ABB Master®

AMIS and Kermit

User's Guide

Use of **DANGER**, **WARNING**, **CAUTION**, and **NOTE**

This publication includes, **DANGER**, **WARNING**, **CAUTION**, and **NOTE** information where appropriate to point out safety related or other important information.

DANGER	Hazards which could result in severe personal injury or death
WARNING	Hazards which could result in personal injury
CAUTION	Hazards which could result in equipment or property damage
NOTE	Alerts user to pertinent facts and conditions

Although **DANGER** and **WARNING** hazards are related to personal injury, and **CAUTION** hazards are associated with equipment or property damage, it should be understood that operation of damaged equipment could, under certain operational conditions, result in degraded process performance leading to personal injury or death. Therefore, comply fully with all **DANGER**, **WARNING**, and **CAUTION** notices.

TRADEMARKS

Advant and AdvaBuild are registered trademarks of ABB Process Automation Corporation, USA

MasterAid is a registered trademark of ABB Asea Brown Boveri LTD, Switzerland

Motorola is a registered trademark of MOTOROLA Inc.

MS-DOS is a registered trademark of Microsoft Corporation

VT100 and VAX/VMS are registered trademarks of Digital Equipment Corporation, USA

NOTICE

The information in this document is subject to change without notice and should not be construed as a commitment by ABB Industrial Systems AB. ABB Industrial Systems AB assumes no responsibility for any errors that may appear in this document.

In no event shall ABB Industrial Systems AB be liable for direct, indirect, special, incidental or consequential damages of any nature or kind arising from the use of this document, nor shall ABB Industrial Systems AB be liable for incidental or consequential damages arising from use of any software or hardware described in this document.

This document and parts thereof must not be reproduced or copied without ABB Industrial Systems AB's written permission, and the contents thereof must not be imparted to a third party nor be used for any unauthorized purpose.

The software described in this document is furnished under a license and may be used, copied, or disclosed only in accordance with the terms of such license.

Valid for

AdvaBuild On-line Builder-FP*1.4

MasterAid 720*4.2 (KERMIT chapter only)

Copyright © ABB Industrial Systems AB 1994

TABLE OF CONTENTS

Chapter 1 - Introduction

1.1 General Information 1-1
1.2 Manual Organization 1-1
1.3 Conventions 1-1
1.4 Related Documents 1-2
1.5 Release History 1-2
1.6 Terminology 1-2
  1.6.1 Terminology used within AdvaBuild On-Line Builder 1-2
    1.6.1.1 Volumes and Segments 1-2
    1.6.1.2 Wildcards 1-3
    1.6.1.3 Default Volume and Segment Names 1-4
  1.6.2 Terminology used within MasterAid 720 1-4
1.7 Equipment Requirements 1-4
1.8 Product Overview 1-5
1.9 User Interface 1-5

Chapter 2 - Editing with AMIS - Introduction

2.1 Introduction 2-1
  2.1.1 AMIS 2-1
  2.1.2 Two Alternative Operating Modes 2-1
2.2 Terms 2-1
  2.2.1 Buffer 2-2
  2.2.2 Work Point 2-2
  2.2.3 Window 2-2
  2.2.4 Mode Line 2-3
  2.2.5 Echo Line 2-3

Chapter 3 - Editing with AMIS - Function keys (EDIT)

3.1 Introduction 3-1
3.2 Basic Commands 3-1
  3.2.1 Start Editing 3-1
  3.2.2 Inserting Text 3-1
  3.2.3 Writing Commands 3-2
  3.2.4 Moving the Work Point 3-4
  3.2.5 Deleting Text 3-4
  3.2.6 Segments 3-5
  3.2.7 Escaping Difficult Situations 3-6
  3.2.8 Escaping AMIS 3-6

CONTENTS (continued)

3.3 Start of Edit 3-6
  3.3.1 Semantics 3-6
  3.3.2 Internal Segments 3-7
  3.3.3 Backup Segments 3-7
3.4 Mark and Region 3-8
  3.4.1 Moving the Mark 3-8
  3.4.2 Setting the Region 3-8
  3.4.3 Other Uses of the Mark 3-8
  3.4.4 Old Marks 3-9
3.5 Erasing and Moving Text 3-9
  3.5.1 Erasure 3-9
  3.5.2 Delete 3-9
  3.5.3 Kill 3-9
  3.5.4 Moving and Copying Text 3-10
  3.5.5 Q-registers 3-10
3.6 Search 3-10
  3.6.1 Incremental Search 3-10
  3.6.2 Subcommands 3-11
  3.6.3 If a String is Not Found 3-11
3.7 Replacing Text 3-12
3.8 Handling segments 3-13

Chapter 4 - Editing with AMIS - Control Characters

4.1 Introduction 4-1
4.2 Terms 4-1
  4.2.1 Overwritten Messages 4-1
4.3 Basic Commands 4-1
  4.3.1 Starting AMIS 4-1
  4.3.2 Inserting Text 4-2
  4.3.3 Writing Commands 4-2
  4.3.4 Moving the Work Point 4-3
  4.3.5 Deleting Text 4-3
  4.3.6 Segments 4-4
  4.3.7 Escaping Difficult Situations 4-5
  4.3.8 Escaping AMIS 4-5
  4.3.9 Help Commands 4-5
  4.3.10 Using Blank Lines 4-6

CONTENTS (continued)

4.4 Start of AMIS 4-7
  4.4.1 Semantics 4-7
  4.4.2 Internal Segments 4-7
  4.4.3 Backup Segments 4-7
  4.4.4 Editing with Function Keys 4-8
4.5 Numeric Arguments 4-8
4.6 How Commands are Written 4-9
  4.6.1 Expansion to More Commands 4-9
  4.6.2 Named Commands 4-10
4.7 How to Escape 4-11
  4.7.1 Escaping AMIS with <CTRL-X> <CTRL-Z> 4-11
  4.7.2 Interruption With <CTRL-G> 4-12
  4.7.3 Reading the Mode Line 4-12
  4.7.4 Recursive Calling of AMIS 4-12
4.8 Help and Documentation Commands 4-12
  4.8.1 Command Description from Character (C) 4-13
  4.8.2 Command/Variable Description from Name (D) 4-13
  4.8.3 Command Character from Name (W) 4-13
  4.8.4 Command Name from Text String (A) 4-13
  4.8.5 Information Regarding Help Character (<CTRL-_>) 4-14
4.9 Moving the Work Point 4-14
  4.9.1 Moving Short Distances 4-14
  4.9.2 Moving Relative to Sentences and Characters 4-15
  4.9.3 Moving Longer Distances 4-15
4.10 Mark and Region 4-15
  4.10.1 Moving the Mark 4-16
  4.10.2 Setting the Region 4-16
  4.10.3 Other Uses of the Mark 4-16
4.11 Erasing and Moving Text 4-17
  4.11.1 Erasure 4-17
  4.11.2 Delete 4-17
  4.11.3 Kill 4-17
  4.11.4 Moving and Copying Text 4-18
  4.11.5 Q-registers 4-18
  4.11.6 Switch-Places Commands 4-19
4.12 Search 4-20
  4.12.1 Incremental Search 4-20
  4.12.2 Subcommands 4-21

CONTENTS (continued)

4.12.3 If a String is not Found 4-21
4.12.4 Single Search 4-21
4.13 Replacing Text 4-22
4.14 Major Modes and Minor Modes 4-22
4.14.1 Major Modes 4-23
4.14.1.1 Fundamental Mode 4-23
4.14.1.2 Text Mode 4-23
4.14.1.3 Keypad Mode 4-23
4.14.2 Minor Modes 4-23
4.14.2.1 Overwriting 4-24
4.14.2.2 Automatic Line Change 4-24
4.15 Special Text Commands 4-24
4.15.1 Switching Between Upper- and Lower- Case Letters 4-25
4.15.2 Indented Text 4-25
4.15.3 Margin Settings 4-26
4.16 Multiple Buffers 4-27
4.17 Handling Segments and Buffers 4-28
4.18 Two Windows 4-30
4.19 Command Macros 4-31
4.20 Function Keys 4-32
4.21 List of Selected Commands 4-34

Chapter 5 - KERMIT

5.1 Introduction 5-1
5.2 Transferring Files 5-2
5.2.1 Setting up the Connection 5-2
5.2.1.1 Preparing the Advant Station 100 Series Engineering Station 5-2
5.2.1.2 Preparing the MasterAid 720 5-3
5.2.2 Setting the File Type 5-3
5.2.3 Transferring Files from AS 100ES to MasterAid 720 5-4
5.2.3.1 Transferring Single Files 5-4
5.2.3.2 Transferring Multiple Files 5-5
5.2.4 Transferring Files from MasterAid 720 to AS 100ES 5-5
5.2.4.1 Transferring Single Files 5-5
5.2.4.2 Transferring Multiple Files 5-5
5.2.5 Closing the Connection 5-6
5.2.5.1 Using the AS 100ES as a Terminal 5-6
5.2.6 Examples 5-7
5.2.6.1 Example 1 5-7
5.2.6.2 Example 2 5-8
5.2.6.3 Example 3 5-9

CONTENTS (continued)

5.3 File Types 5-11
  5.3.1 ASCII (text) Files........ 5-11
  5.3.2 Binary Files........ 5-12
  5.3.3 Load Module Files........ 5-13
  5.3.4 Copying AS 100ES Volumes 5-13
  5.3.5 Backing up AS 100ES Files to a Foreign System 5-13
5.4 Description of Commands 5-14
  5.4.1 Command Summary 5-14
  5.4.2 Detailed Command Descriptions........ 5-15
    5.4.2.1 Command Syntax........ 5-15
    5.4.2.2 Command Entry 5-15
    5.4.2.3 Bye 5-16
    5.4.2.4 Connect 5-16
    5.4.2.5 CWD 5-17
    5.4.2.6 Exit 5-17
    5.4.2.7 Finish........ 5-17
    5.4.2.8 Get........ 5-18
    5.4.2.9 Local........ 5-18
    5.4.2.10 Local Directory 5-19
    5.4.2.11 Local Space 5-19
    5.4.2.12 Local Status........ 5-19
    5.4.2.13 Local Type 5-19
    5.4.2.14 Logout 5-20
    5.4.2.15 Quit........ 5-20
    5.4.2.16 Receive........ 5-20
    5.4.2.17 Remote 5-21
    5.4.2.18 Send........ 5-23
    5.4.2.19 Server 5-23
    5.4.2.20 Set........ 5-24
    5.4.2.21 <CTRL-BREAK> 5-26
    5.4.2.22 <CTRL-F4> <C> 5-27
    5.4.2.23 Show........ 5-27
    5.4.2.24 Statistics 5-28
    5.4.2.25 Take........ 5-28
5.5 Cables for KERMIT Connections........ 5-28

INDEX

Chapter 1 Introduction

1.1 General Information

This guide describes the usage of two utility programs delivered together with AdvaBuild On-line Builder for Advant Station 100 Series Engineering Stations: the text editor AMIS and the data exchange program KERMIT. KERMIT is improved to transfer (besides ordinary text files) binary load modules between AS 100ES and MasterAid 720. It is part of AdvaBuild On-line Builder for AS 100ES as well as MasterAid 720 software.

1.2 Manual Organization

The text editor AMIS is explained from Chapter 2 to Chapter 4. Chapter 2 gives an introduction to editing with AMIS. How to work with function keys is explained in Chapter 3. This is the usual way of working with AMIS. More advanced editing is possible with control characters. How you perform editing with control characters is described in Chapter 4. These chapters are valid for AdvaBuild On-line Builder for Advant Station 100 Series Engineering Station only.

Chapter 5 deals with KERMIT. This chapter is valid for both, MasterAid 720 software and AdvaBuild On-line Builder for Advant Station 100 Series Engineering Station.

1.3 Conventions

Keyboard Interaction

When you read	**You shall**
<F4>	Press the key labeled **F4**.
<Ctrl-Break>	Press the **Ctrl** key while pressing the key labeled **Break**.

Syntax

In this manual the syntax of each command is described in the form of the command name, followed by the arguments that are to be specified. The arguments are separated with symbols such as comma “,”, slash “/”, semicolon “;” or right angle bracket “ >”. To carry out a command, enter the command syntax and press the **Enter** key.

- **Bold** strings are the command names and arguments; they must be typed as they appear.
- *Italic* strings represent substitutable arguments.
- Brackets “[ ]” around an argument name indicate that the argument is optional.
- `Courier` font is used for output displayed on the screen, or input on the terminal.

Arguments, to be used alternatively are indicated by a vertical bar (in some syntax descriptions alternative choices are represented by an italic string and a separate explanation of its substitutions).

Examples:

```
LTREE item_des[;number_of_elements][/mode]
LDBD SEL=select[=start][,interv][,gap][/XREF[;PCm[>PCn]]]
```

The following conventions are used in examples of screen dialogs:

- Bold strings represents input from the USER
- Normal strings represents dialog text from the target system or the engineering station
- Courier font is used

1.4 Related Documents

Related documentation, including those referred to in this document are listed below.

Table 1-1. Related Documentation

Manual	Description
Advant Station 100 Series Engineering Station User's Guide	Contains descriptions of AS 100ES and their functions:
AdvaBuild On-line Builder Reference Manual	Describes all commands used in AS 100ES, AS 500ES and AS 500OS. Command sets and working methods are described as well as command file handling aspects.
MasterAid 720 User's Manual 2	General description how to work with MasterAid 720; part of 4 binders delivered together with MasterAid 720

1.5 Release History

First edition.

1.6 Terminology

1.6.1 Terminology used within AdvaBuild On-Line Builder

1.6.1.1 Volumes and Segments

The AdvaBuild On-line Builder has access to two volumes, SRCE and USER. On the engineering station a set of these two volumes exists in order to store any target specific data for each defined target. Depending on the target selection on the engineering station the corresponding set of volumes (SRCE and USER) are available and refer to the directories within the directory structure of the selected target.

It is recommended to use the SRCE volume, which is the default volume when working with the AdvaBuild On-line Builder rather then the USER volume. The USER volume contains different segments with various application data for internal use of the AdvaBuild On-line Builder.

Each volume may contain a number of segments. A segment is a certain quantity of data or program code which is grouped together. A segment is handled as a unit by the mass storage and has a segment name, which corresponds with a file on the according directory.

A segment has the following segment identifier, that contain a colon and a period as separators:

`volume``:`*`segment.extension`*

The following characters are valid (other characters may appear in volume names given by the system):

volume	= 4 ASCII characters; A-Z, a-z, 0-9
segment	= 1-8 ASCII characters; A-Z, a-z, 0-9
extension	= 2 ASCII characters; A-Z, a-z, 0-9

Example:

```
USER:ABCD.AA
```

In syntax definitions the extension of a segment is not mentioned, but is a part of “segment”.

Within AMIS “file” is sometimes used in messages instead of “segment”. In KERMIT “directories” correspond to “volume” and “files” to “segments”.

1.6.1.2 Wildcards

A wildcard is a character which stands for any other character.

Wildcard characters are accepted by many commands when handling segment identifiers. Two different wildcard characters are accepted:

%	(percent) means that any single character can occur in this position.
*	(star) works in the same manner as %, but with the difference that the star applies for all subsequent positions. Applicable within a segment name or within an extension.

Examples

Example 1:

```
#LDIR USER:A*.%1
```

Lists the segments from USER volume which have segment names beginning with A, and extensions ending with 1.

Example 2:

```
#LDIR :A%%.*
```

Lists all segments whose segment name consists of three characters, the first of which is A. The extension can be optional.

1.6.1.3 Default Volume and Segment Names

Many commands accept default volume and segment names. However, separators between the parameters must always be written (“:” and “>”).

The volume and segment names are stored S1 and S2, which retain the names in the segment identifiers. A command which contains only one identifier uses S2. A command which contains two identifiers uses S1 for the first identifier and S2 for the second.

Each time an identifier is given in a command, S2 is updated (as is S1, if two identifiers are given). However, there is an exception to this:

```
LDIR volume:
```

is synonymous with

```
LDIR volume:*.*
```

which does not update S2.

Example:

```
COPY USER:S*.CP>SRCE:*.*
```

copies the segments from volume user to volume SRCE, whose segment names begin with S and whose extensions = CP.

Copying can now be checked with:

```
COMPARE : > :
```

A list of all of the segments in volume SRCE (S2) is now obtained with:

```
LDIR :
```

1.6.2 Terminology used within MasterAid 720

Volumes and segments within AdvaBuild On-line Builder corresponds directly to directories and file names in MasterAid 720. For directories and file names the rules of VAX/VMS apply. Please, consult the VAX/VMS documentation.

1.7 Equipment Requirements

For working with AMIS it is necessary to have AS 100ES as delivered by ABB Process Automation or an equivalent IBM PC/AT equipped with Advant Station 100 Series Engineering Kit. For demands on an appropriate personal computer see Advant Station 100 Series Engineering Station User’s Guide.

For transferring data between Advant Station 100 Series Engineering Station and MasterAid 720 with KERMIT the following equipment is necessary:

- An engineering station as described above
- A MasterAid 720 installation on a VAX/VMS system

1.8 Product Overview

For engineering stations of type Advant Station 100 Series, AMIS and KERMIT are part of the AdvaBuild On-line Builder function. For MasterAid 720 KERMIT is part of the standard delivery.

1.9 User Interface

The user interface used on each platform is described in the documents listed in Section 1.4 Related Documents.

Chapter 2 Editing with AMIS - Introduction

2.1 Introduction

The description of the AMIS text editor is divided into three chapters. Chapter 2 Editing with AMIS - Introduction is a general introduction. Chapter 3 Editing with AMIS - Function keys (EDIT) and Chapter 4 Editing with AMIS - Control Characters describe two alternative operating modes for the text editor.

Read “Start of Edit” in Chapter 3 and “Start of AMIS “ in Chapter 4. You are now ready to begin using AMIS in practice.

The remaining subsections can be perused in order to gain an orientational understanding of the described commands and their use. Deeper study of these sections can take place subsequently, as the new commands are used.

Chapter 3 and Chapter 4 conclude with lists of the commands.

AMIS can be used to edit compressed segments stored on volumes SRCE and USER. Compressed segments don’t use the standard ASCII representation, but a compressed one. The segments can be distinguished from normal DOS files by the two-character extension.

2.1.1 AMIS

AMIS is an advanced screen-oriented text editor. AMIS consists of a program which is used to create, read and edit text segments. The text is displayed on the screen during editing. Entered changes of the text are displayed on the screen immediately.

2.1.2 Two Alternative Operating Modes

The AMIS editor has two alternative operating modes:

- A simpler operating mode in which all of the commands are given by means of the function keys (described in Chapter 3).
- A more advanced mode in which the commands are given by means of control characters (described in Chapter 4).

The function keys include the keyboard’s function keys as well as the keys used to move the cursor (arrows). This alternative contains only some of the AMIS functions. Nevertheless, these are sufficient for ordinary editing.

The control characters are generated by simultaneously pressing the **Ctrl** key and the key of a character, usually a letter. All AMIS functions are available in this mode.

2.2 Terms

This section defines some of the fundamental terms and principles which are used when working with AMIS.

The first term considered is “buffer”. We will then describe the work point and how it is used when working within the buffer. Finally, a description of how the various parts of the screen are used is presented. The term volume can be either USER or SRCE volume.

2.2.1 Buffer

“Buffer” is the name given to the space in the computer’s memory in which the text is stored while it is being edited.

When you start AMIS to modify an existing segment, a copy of the segment’s contents is written into the buffer. All changes you perform will be stored inside this buffer. After editing, the contents of the buffer is written back into the segment.

The entire procedure is driven by your commands given from the keyboard. AMIS automatically keeps track of which segments are in the buffer.

2.2.2 Work Point

The “work point” is the actual position within the buffer where changes will be performed. Most commands perform their operations at the work point; new characters are entered there and characters are removed from this same point during erasure.

The work point is moved within the buffer by means of special commands. The work point is always located between two characters in the buffer (or before the first, or after the final character).

2.2.3 Window

During editing with AMIS, the text is displayed on the screen. That portion of the screen which displays the text is called the “window” (a window into a (restricted) area of the buffer).

The lowest lines on the screen (outside of the window) are used for other purposes (described in the following sections).

The window always displays that part of the buffer in which the work point is located. If the work point is located on the window’s upper or lower edge and a command to move the work point towards the window’s edge is given, the window’s position in the buffer will also be moved. In this way, the work point is always kept within the window.

When AMIS is awaiting input from the keyboard, the cursor (a blinking rectangle or line) is positioned immediately after the work point (work point between two characters). The cursor indicates the position of the work point.

In this description an underline character (“_”) will be used to indicate the work point’s position. This symbol corresponds to the system’s blinking cursor. The work point is located on the left edge of the character.

Sometimes lines in the buffer are so long that they cannot fit on the screen. AMIS splits such lines into two screen lines, and marks the split by means of an exclamation mark (“!”) on the right edge. This applies only to the display in the window. The lines can be of any length in the buffer. In the segment, lines longer than 132 characters are truncated.

2.2.4 Mode Line

The “mode line” is the line located immediately below the window on the screen. It usually begins with “AMIS”, and it contains important information about the buffer in use.

A typical mode line can be:

```
AMIS (Keypad) Main: volu:segname.ex --17%-- *
```

AMIS	indicates that the AMIS program is being run. Another name may sometimes occupy this position, and in such cases, the AMIS commands will not function as usual.
(Keypad)	indicates that the editor is working with function keys and was started with the EDIT command
Main:	name of the buffer presented in the window.
volu:segname.ex	indicates the name of the segment in the buffer. If the buffer contents is written back to the segment without a new segment name, the previous name will automatically be used again.
--17%--	indicates the location of the window in the buffer. 17% means that 17 percent of the entire text is above the window.
--TOP--	indicates that the window is located at the beginning of the buffer.
--BOT--	indicates that the window is located at the bottom of the buffer. If the text in the buffer is so short that the entire text fits within the window, this information is omitted entirely.
*	means that some change has been made in the text (after it was written into the buffer).

2.2.5 Echo Line

An “echo line” is located below the mode line. Short messages and questions to you are written on this line. In some cases, the indicated commands are also presented on this line.

Chapter 3 Editing with AMIS - Function keys (EDIT)

3.1 Introduction

The following description applies to the simpler of the two editing modes. Editing is performed using the function keys.

3.2 Basic Commands

This section describes some of the most common commands: inserting text, deleting text, moving in the buffer and writing the results into a segment.

This section contains many references to sections which come later in the description. This is due to the fact that important sections are expanded upon in detail later in the description.

It is to the user’s advantage if the description can be read in conjunction with editing on the Advant Station 100 Series Engineering Station.

3.2.1 Start Editing

The EDIT command is used to start editing with the function keys (described in Section 3.3 Start of Edit). The system also reads the indicated segment into the buffer during startup.

3.2.2 Inserting Text

No special command is needed to insert text at the work point (no “Insert Mode” such as some editors have). All typed characters are stored directly in the buffer. The text which is written is always located at the work point.

Example:

```
Woozle
```

The work point is located in front of the “z” (the cursor on the screen will blink at “z”). If <**A**> is pressed down, the result will be:

```
WooAzle
```

The work point retains its position in front of the <**z**>.

Line changes are made by pressing <**ENTER**>. This function is used to create new blank lines (when the work point is at the end of a line), as well as in dividing lines (when the work point is within the line). If <**ENTER**> is pressed in the previous example, the result will be:

```
WooA
zle
```

Incorrectly written characters are deleted by the **DEL** key (delete). <**DEL**> deletes the character located immediately to the left of the work point.

Line changes can also be cancelled by using the <**DEL**> key. If the work point is located at the beginning of a line (immediately after a line change), <**DEL**> will cause the line change to be cancelled (the line will be linked to the preceding line).

3.2.3 Writing Commands

The commands can be entered by means of both, the function keys and the special keys on the keypad.To use the keypad commands the NUMLOCK key shall be active.

* End of Segment	- Forward Search	+ Unkill
7 Backward Word	8 Forward Word	9 Backward Search
4 Beginning of Line	5 End of Line	6 Beginning of Line
1 Previous Screen	2 Next Screen	3 Exch Point and Mark
0 Extended Command		. Set Mark

Figure 3-1. Keypad

F1 SAVE Segment	F2 Delete Character	F3 Kill Line	F4 Kill Region

Figure 3-2. Function Keys

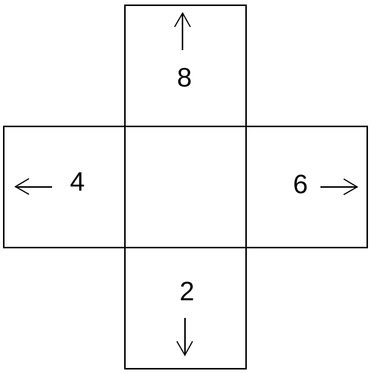

Figure 3-3. Keys for moving the Cursor

When NUMLOCK is active, use the arrow keys together with the SHIFT key to move the cursor. When NUMLOCK is not active, the arrow keys work directly.

The text given in Figure 3-1 for each key is not actually printed on the keyboard's keys. This text is included in the figure to show the functions which are available, and their locations.

The command <**1**> signifies that the function key marked "1" has been depressed, thereby activating the "Previous Screen" command. One must differentiate between the function key marked "1" and number key "1" (the latter applies to the character "1").

The text editor is capable of many more functions than those which are activated directly by pressing the function keys. These functions are accessed by pressing <0>, which displays the following text:

Command:

The user gives the command name and concludes with <**ENTER**>.

The command name often consists of several words. Common commands include:

- Exit
- Write segment
- Insert segment
- Query replace

Additional commands are listed in summary 2, at the end of this chapter.

It is permissible to abbreviate command names word for word, as long as they remain unambiguous. When words are abbreviated, AMIS automatically fills in the missing letters when the space key is depressed.

Named commands are designated in the description as shown in the following example:

```
<0>
Command: Exit <ENTER>
```

3.2.4 Moving the Work Point

The following commands are used to move the work point:

<→>	Moves the work point one increment forward
<←>	Moves the work point one increment backward
<↓>	Moves the work point down one line
<↑>	Moves the work point up one line
<**8**>	Moves the work point forward one word
<**7**>	Moves the work point backward one word
<**5**>	Moves the work point to the end of the line
<**4**>	Moves the work point to the beginning of the line
<**2**>	Moves forward one screen page
<**1**>	Moves backward one screen page
<**,**>	Goes to the end of the segment
<**6**>	Goes to the beginning of the segment

Assume that the buffer contains the following text:

```
First line
Second line
Last line
```

The work point is placed in front of the letter “n” in the word “second”. Pushing the <→> moves the work point in front of the “d”, <≠≠↑> moves it in front of the “t” in the word “First”, while <**4**> moves it in front of the “S” in “Second”.

In this manner, the work point can be placed anywhere in the buffer.

<**2**> is pressed in order to leaf forward, one screen page at a time. Similarly, <**1**> is used to leaf backward.

3.2.5 Deleting Text

The described ways of moving and entering text, and of deleting individual characters suffice for elementary editing.

Two new deleting functions (in addition to <**DEL**>) are described in this section:

<**DEL**>	Deletes the character in front of the work point.
<**F 2**>	Deletes the character after the work point (delete).
<**F 3**>	Deletes the rest of the line (kill line).

The <**F 2**> and <**DEL**> commands are both used to delete single characters. <DEL> is used mainly to eliminate errors.

Assume that the buffer contains the following text:

```
Woozle
```

The work point is placed in front of the “z”. After the <**DEL**> command is given, the result will be:

```
Wozle
```

If function <**F 2**> is selected instead of <**DEL**>, the result will be:

```
Woole
```

The <**F 3**> command is used to delete an entire line. This command deletes the text located between the work point and the next line change. The “flag” (invisible code) for the line change itself remains. On the other hand, if the work point is located immediately before a line change, then the line change will be deleted (the next line will be linked with the previous line).

Assume that the buffer contains the text:

```
Woozle
Wizzle
```

The work point is placed in front of the first “z” on the first line. After the <**F 3**> command is given, this will be the result:

```
Woo_
Wizzle
```

The work point is now located after the word “Woo”.

If the <**F 3**> command is given again, the result will be:

```
WooWizzle
```

The line change has been deleted. When an entire line is to be removed, the work point is placed at the beginning of the line. <**F 3**> is then pressed twice. The first time deletes the text forward to the line change.

The second depression deletes the line change. Additional functions for deleting, moving and copying sections of text are described further in Section 3.5 Erasing and Moving Text.

3.2.6 Segments

This section describes the manner in which an edited or newly-written text is read into a segment and saved. <**F 1**> writes the text into the segment (save segment).

The depression of <**F 1**> ensures that the text in the buffer is written into the segment. This command is used before AMIS is escaped. It should also be used occasionally while editing in order to prevent the loss of results in case of a malfunction or a power outage.

A new segment is created in precisely the same way. The new segment name is written in using the EDIT command. The system then writes “New segment” on the echo line. When the text is ready to be saved, <**F 1**> is used.

The handling of segments is described in further detail in Section 3.8 Handling segments.

3.2.7 Escaping Difficult Situations

Extremely difficult situations can sometimes occur. Such situations can be caused by a misconception, or by the depression of the wrong key.

One way to escape from such situations is to press <**CTRL-G**>. AMIS will then terminate its current activity and return to interpreting commands in the usual manner.

3.2.8 Escaping AMIS

AMIS is escaped by using the following command:

```
<0> Exit
```

The contents of the buffer should be saved in a segment before this command is given. In practice, the command for escaping AMIS should be considered as:

```
<F 1> <0> Exit
```

If the <**F 1**> command is not given, the system will display the following question on the screen:

```
Modified buffers exist. Quit (Y or N)?
```

The answer “N” will return the system to editing mode, from which the “<**F 1**> <**0**> **Exit**” command can be given. If the answer “Y” is given, AMIS will be escaped and the contents of the buffer will not be saved.

3.3 Start of Edit

This section describes startup and editing with function keys.

Syntax:

EDIT [*volume*]:*segment.extension*

3.3.1 Semantics

The EDIT command starts AMIS, thereby starting the editing of the segment specified. If the specified segment already exists, it will be read into the work buffer. Otherwise, the editor will start with an empty buffer.

If no volume name is specified, the system will use the last volume name specified.

Example:

```
*EDIT SRCE:SEGMENT.SA
```

Starts editing of the segment SRCE:SEGMENT.SA.

3.3.2 Internal Segments

When the editor starts, an internal segment is created. This segment is used to store the text during editing.

This internal segment is created on the same volume as the segment (to be edited) is located on.

When the editor is started without a segment name having been indicated, the system will use the last volume name specified. If no volume name was specified earlier, AMIS will ask for the name before editing can begin.

There must be space available on the volume for the internal segment. Allowance must also be made for the addition of more text, which will make the internal segment larger than the original segment.

The internal segment is deleted when AMIS is escaped. However, it can remain if the editor has been escaped by means of “abort”. There is no way to recreate a usable segment from this internal segment alone. The segment, which is called AMISWORK.ET, should therefore be deleted.

3.3.3 Backup Segments

While editing an existing segment, one normally keeps the original segment. The primary reason for this is to ensure that the original segment is protected in case a malfunction (such as a power outage) occurs during editing, or while the edited segment is being inputted. The original segment can also be useful in cases where faulty editing has been carried out.

The original segment automatically receives the extension BP (backup) when the new edited segment is inputted.

If the backup segment (original segment) is to be used instead of the edited segment, proceed as follows.

1. Delete the edited segment:

   ```
   *DELETE volu:segname.ex
   ```

2. Rename the backup segment:

   ```
   *RENAME volu:segname.BP>*.ex
   ```

However, there are commands which write the edited segment over the original segment. In such cases, the amount of space used up on the diskette is not as great. However, if such a procedure is unsuccessful, both the original segment and the edited segment may be lost.

The commands for using segments are described in Section 3.8 Handling segments.

3.4 Mark and Region

Many commands operate within a region in the buffer (not just at the work point). Starting and stopping positions must be specified for such operations. One of these positions will be indicated by the work point, while the other will be indicated by the “mark”. The text located between the mark and the work point comprises the “region”. Either the mark or the work point can be used to indicate the starting position (with respect to the stopping position).

3.4.1 Moving the Mark

The mark works like another work point. The mark has no symbol and therefore cannot be seen. For this reason, it is advisable to place the mark immediately before using it, so that the user has the placement memorized.

The position of the mark is set by moving the work point to the desired position, and pressing the <**.**> key. The positions of the mark and the work point can be exchanged by pressing <**3**>. Thus, the following commands apply to the mark:

<**.**>	Sets the mark at the work point
<**3**>	Switches the mark and the work point

3.4.2 Setting the Region

A region is specified as follows: Place the work point at one of the desired region’s limit points and give the command <**.**>. The mark is now set. Move the work point to the region’s other limit point. The region is now set.

Check the region by depressing <**3**>. The work point will alternate between the starting and stopping positions each time <**3**> is depressed.

3.4.3 Other Uses of the Mark

There are commands which move the work point far away from its initial position. Some of these commands automatically set the mark at the work point’s initial position so that it can be found again easily. Typical examples of such commands are the search command <**-**> and <**9**>. These commands set the mark at the work point’s initial position if the search procedure moves the work point more than 500 characters.

One type of command sets a region around an inserted section of text. This is true of commands which insert sections of text when, for example, they are being moved or copied in from another segment. This function makes it very easy to continue work on the text section in question.

3.4.4 Old Marks

If a section of text is inserted or deleted in front of the mark, the mark will be moved in relation to the text. This is due to the fact that the mark maintains its distance from the start of the buffer. The mark should therefore always be set immediately before it is to be used.

3.5 Erasing and Moving Text

This section described the commands used to erase and move text. These functions are described together because they make use of the same commands, to a certain extent.

3.5.1 Erasure

Two types of erase commands are found in AMIS. One type includes those commands which are used to remove individual characters or blank lines. The other type consists of those commands which remove entire words, lines, sentences or regions. The first type of commands are called Delete commands, while the second type are called Kill commands. Text which is removed by Delete commands is lost entirely. On the other hand, text removed by Kill commands is saved in a special erase buffer. This buffer can then be used as a normal text segment, i.e., it can be copied and inserted wherever one wishes. Using the erase buffer provides a convenient means of moving and copying text.

3.5.2 Delete

These commands remove characters without saving them:

<**F 2**>	Removes the character to the right of the work point
<**DEL**>	Removes the character to the left of the work point

3.5.3 Kill

These commands remove text, but they also save the text in an erase buffer. If several Kill command operations are carried out in sequence, the combined text which has been removed will be saved.

<**F 3**>	Removes the rest of the line
<**F 4**>	Removes a region

An entire line can be removed by placing the work point at the start of the line and then giving the command <**F 3**>. The line will become empty up to the line change. Another depression of <**F 3**> will remove the remaining line change.

A region is removed by first specifying its extent by using the mark and the work point, and by then giving the command <**F 4**>.

3.5.4 Moving and Copying Text

If a text is to be moved (to one or more locations), it is first placed in the erase buffer through the use of a Kill command. The work point is then moved to the desired text location. The depression of <**+**> inserts the text into the desired location. Repeated depression of <**+**> duplicates the text in the locations selected. In this manner, words, lines, sentences and regions can be moved and duplicated.

<**+**>	Inserts the text from the erase buffer

3.5.5 Q-registers

Use of the erase buffer entails a temporary storage of the removed text. More permanent storage is available through the use of the Q-registers. There are many Q-registers with different names, which consist of a single letter A-Z.

<**0**>**`Put Q-reg`**	Copies the region to the Q-register
<**0**>**`Get Q-reg`**	Inserts text from the Q-register

Both functions demand the name of the desired Q-register.

Put Q-reg means that a copy of the area specified by the mark and the work point will be made. The copy is input into the selected Q-register.

Get Q-reg means that the text in the indicated Q-register will be inserted at the position where the work point is located.

3.6 Search

As the text increases in size, it becomes more difficult to find things within it. AMIS has commands for searching both forward and backward, thereby facilitating search procedures.

3.6.1 Incremental Search

<**-**>	Incremental search forward.
<**9**>	Incremental search backward.

The command <**-**> starts an incremental forward search process. The command <**9**> starts a corresponding backward search process.

Once the search process has begun, it is possible to write in the text which is being searched for. It is also possible to give subcommands to control the search.

The work point’s prospective placement is presented in the window at the same time the search text is being written in and the subcommands are given.

The search process is terminated by pressing <**ESC**>. The work point will then move to its new location.

This search process is called an “incremental search”.

The text “I-Search:” or “Reverse I-Search:” is placed on the echo line while the search is in progress. The search text is displayed after the colon. Text which is being inserted is harbored in this text string. The work point moves forward to the first occurrence of the corresponding text string. The characters in the text string are not echoed directly, but rather presented on the echo line in the order in which the editor finds the various characters.

No differentiation is made between upper- and lower-case letters in the search process.

3.6.2 Subcommands

Subcommands are often given in the form of control characters. Control characters are generated by holding down the <**CTRL**> key while simultaneously writing a character, usually a letter. Other control characters which have their own keys are <**DEL**> and <**ESC**>.

The following commands can be given during a search:

<**CTRL-S**>	Searches forward for next occurrence.
<**CTRL-R**>	Searches backward for next occurrence.
<**DEL**>	Erases last character written in search string.
<**ESC**>	Terminates search.
<**CTRL-G**>	Terminates search. Returns to initial position.

If a search proves unsuccessful, the search string can be expanded. Alternatively, <**CTRL-S**> or <**CTRL-R**> can be pressed repeatedly to search for the next occurrence of the same string.

If the search string specified last is to be used again, it can be recalled by pressing <**CTRL-S**> or <**CTRL-R**>. This applies only when no search string has been written in.

If the search moves the work point more than 500 characters, the mark will automatically be set to the starting position. In such cases, “***Mark Set***” will be indicated on the echo line. The mark is used to facilitate return to the starting position.

3.6.3 If a String is Not Found

If AMIS does not find the correct string, “Failing I-Search:” or “Failing Reverse I-Search:” will be displayed on the echo line. If the search is to be concluded at this point, this is done by pressing <**ESC**>.

An unsuccessful search may be due to an incorrectly written search string. Deletion is carried out character by character, using <**DEL**>.

If a forward search is unsuccessful, it may be due to the fact that the section being sought is located back in the text. Continue the search with <**CTRL-R**>.

If a search is terminated with <**CTRL-G**>, everything will be returned to the pre-search positions. The work point will be moved back to the starting position.

3.7 Replacing Text

It is sometimes necessary to replace one text string with another throughout the text, or just at certain points. In either case, the Query Replace command can be used:

<0> Query Replace Selective replacement.

This command causes two questions to be asked. The first question concerns the string which is to be deleted. The second question concerns which string is to replace the deleted string.

After these questions have been answered, AMIS will perform a search from the work point to the end of the buffer. When a string which corresponds to the string which is to be deleted is found, the work point moves to it. At this point, AMIS will wait for one of the following responses:

<**SPACE**>	Replace the text string. Search further.
<**DEL**>	Do not replace the text string. Search further.
<**ESC**>	Terminate the activity.
<**.**>	Replace the text string, then terminate the activity.
<**!**>	Replace the text string here and in the rest of the buffer.
<**,**>	Replace the text string. Search no further.

If replacement is to occur throughout the entire buffer, the work point is placed at the beginning of the buffer. This is done before the selective replacement begins. While the activity is going on, <**SPACE**> OR <**DEL**> is pressed to replace or skip over each string.

If the replacement is to occur throughout, an exclamation mark is used as the response.

No differentiation is made between upper- and lower-case letters in the search process.

Upper- and lower-case letters for the replacement string are determined as this string is written in. This disposition is retained upon replacement, since it agrees with the letter disposition of the replaced string.

3.8 Handling segments

Earlier sections in this description have superficially touched on how to handle segments. A more thorough study of the handling of segments is presented in this section.

Table 3-1. Handling of Segments

Copying and moving from segment to buffer:	
<0> Insert File	Contents of segment written into buffer
Copying and moving from buffer to segment:	
<F1>	Contents of buffer written into opened segment
<0> Write File	Contents of buffer written into opened segment
<0> Write Region	A region is written into opened segment
Original segments not saved for indicated commands:	
<0> No Backup Save File	Applies to: <F1>
<0> No Backup Write File	Applies to: <0> Write Region
Display contents of another segment:	
<0> View File	Contents of given segment will be listed on screen

<0> Insert File causes a text which is located in a segment to be copied and inserted into the work buffer. The text is inserted at the position where the work point is located. This command presents the following question:

```
Insert File:
```

The desired segment name is indicated. If the volume name is omitted, the system will use the last name indicated.

<F1> is used to write the buffer contents into the segment which is already opened. The opened segment’s name is indicated on the mode line. This command is used after editing is completed, and sometimes during editing in order to ensure that the editing which has been completed will be saved in case of mechanical malfunction or a power outage.

<0> Write File is used instead of <**F 1**> if the buffer’s contents are to be written into another segment instead of the opened segment. This command presents the following question:

```
Write File:
```

The name of the segment into which the contents are to be written is given as the answer.

<0> Write Region is used to write in a region in the segment which has already been opened. The handling procedure is the same as for <**F 1**>, with the exception that the region in question must be specified by the mark and the work point before the command is given.

<0> No Backup Save File, <0> No Backup Write File and **<0> No Backup Write Region**. These commands are used when the original segment is not to be saved.

The original segment is normally saved during editing. The edited text is placed in a new segment with the same identifier as before. The original segment retains the same name, but the extension BP (backup) is added.

When these commands are used, the edited text is written "over" the original text in the original segment. For example, these commands can be used when the available space on the diskette does not allow for both the original and edited segments.

<0> View File. This command lists the contents of a given segment on screen. Press <SPACE> to initiate scrolling to next page. Any other command will terminate displaying the contents on screen.

Summary 1: Function Keys (with NUMLOCK active)

F1 SAVE Segment	F2 Delete Character	F3 Kill Line	F4 Kill Region

Figure 3-4. Function Keys (with NUMLOCK active)

* End of Segment	- Forward Search	+ Unkill
7 Backward Word	8 Forward Word	9 Backward Search
4 Beginning of Line	5 End of Line	6 Beginning of Line
1 Previous Screen	2 Next Screen	3 Exch Point and Mark
0 Extended Command		. Set Mark

Figure 3-5. Keypad

Summary 2: List of Named Commands

When a named command is to be initiated, the <**0**> function key is depressed. This asks the following question:

```
Command:
```

The answer will consist of the command name, followed by <**ENTER**>.

Table 3-2. List of Named Commands.

Command Name	Function Performed
exit	Escape AMIS
query replace	Selective replacement
get Q-reg	Copy region to Q-register
put Q-reg	Insert text from Q-register
view file	Read a segment into the buffer
insert file	Insert text from a segment
write file	Write the buffer contents into a segment
write region	Write the region into a segment
no backup save file	Write the buffer contents into a segment; no backup copy
no backup write region	Write the region into a segment; no backup copy

Chapter 4 Editing with AMIS - Control Characters

4.1 Introduction

The following description concerns the more advanced of the two available editing modes. Editing is performed by means of control characters, but it is also possible to edit using the function keys, as described in Chapter 3 Editing with AMIS - Function keys (EDIT).

4.2 Terms

The terms which are common to both editing modes are described in Chapter 2 Editing with AMIS - Introduction. One additional term is required for editing using the control characters. This term will be presented in the next section.

4.2.1 Overwritten Messages

Short messages are displayed on the echo line. Longer messages and help texts do not fit into this space. For this reason, a special method is used to present them.

This method is called “overwriting”, and it means that the message is written over the window on the screen. The message will disappear if the space key is depressed, or if a command is given.

A message may be so long that even the entire window’s space is insufficient. In such cases, the text --MORE-- is written on the right-hand portion of the mode line to mark the additional text. The hidden portion of the message will then be presented when the space key is depressed. If DELETE or a command is entered instead, the remaining portion of the message will disappear. The word FLUSHED will then be displayed on the right-hand portion of the mode line.

4.3 Basic Commands

This section describes some of the most common commands: inserting text, deleting text, moving text in the buffer and inputting results in a segment.

The section contains many references to sections which come later in the description. This is due to the fact that important sections are expanded upon in detail later in the description.

It is to the user’s advantage if the description can be read in conjunction with editing on the Advant Station 100 Series Engineering Station.

4.3.1 Starting AMIS

The AMIS command is used to start editing with the control characters (or function keys). This command is described further in Section 4.4 Start of AMIS. The start of the editing process also means that the system writes the indicated segment into the buffer. However, it is possible to start without writing in any segments (described in Section 4.4.1 Semantics).

4.3.2 Inserting Text

No special command is needed to insert text at the work point (no “Insert Mode” such as some editors have). All of the usual written characters are stored directly in the buffer.

The text which is written is always located at the work point.

Example:

```
Woozle
```

The work point is located in front of the “z” (the cursor on the screen will blink at “z”). If the “A” key is depressed, the result will be:

```
WooAzle
```

The work point retains its position in front of the “z”.

Line changes are made by pressing the RETURN key <**ENTER**>. This function is used to create new blank lines (when the work point is at the end of a line), as well as to divide lines (when the work point is within the line). If <**ENTER**> is pressed in the previous example, the result will be:

```
WooA
zle
```

Incorrectly written characters are deleted by the DELETE key <**DEL**>. <**DEL**> deletes the character located immediately to the left of the work point.

Line changes can also be cancelled by using the <**DEL**> key. If the work point is located at the beginning of a line (immediately after a line change), <**DEL**> will cause the line change to be cancelled (the line will be linked to the preceding line).

4.3.3 Writing Commands

This section describes how commands are constructed during editing with the control characters.

The editor stores all of the characters which belong to a given text in the work buffer. In order for the editor to differentiate between these characters and the control characters, the latter must be used in combination with the control key <**CTRL**>. Control character commands are initiated by simultaneously pressing down <**CTRL**> and another key. However, some control characters do have their own keys.

The most important of these is ESCAPE, which uses the <**ESC**> key. The two most common types of command are the <**CTRL**> and <**ESC**> commands.

The **CTRL** key functions as a shift key when control commands are in use (writing in one position and controlling in the other).

If the keyboard is equipped with a <**META**> key, this key is used in the same way as the **CTRL** key. The Advant Station 100 Series Engineering Station has no **META** key, and the **META** commands are instead given by pressing <**ESC**>, followed by a character.

Example:

`<CTRL-A>`

The CTRL key and the A key are depressed simultaneously.

`<ESC><A>`

The ESC key is pressed for a moment. The A key is then pressed. (No simultaneous pressing down).

4.3.4 Moving the Work Point

The following commands are used to move the work point:

<**CTRL-F**>	(Forward)	Moving forward one character
<**CTRL-B**>	(Backward)	Moving backward one character
<**CTRL-N**>	(Next)	Moving down one line
<**CTRL-P**>	(Previous)	Moving up one line
<**CTRL-E**>	(End)	Moving to the end of the line
<**CTRL-A**>		Moving to the beginning of the line
<**CTRL-V**>		Moving forward one screen page
<**ESC**><**V**>		Moving backward one screen page

Assume that the buffer contains the following text:

```
First line
Second line
Last line
```

The work point is placed in front of the letter “n” in the word “second”. The command <**CTRL-F**> moves the work point in front of the “d”, <**CTRL-P**> moves it in front of the “t” in the word “First”, while <**CTRL-A**> moves it in front of the “S” in “Second”.

In this manner, the work point can be placed anywhere in the buffer. The <**CTRL-V**> command is used to leaf forward, one screen page at a time. Similarly, <**ESC**><**V**> is used to leaf backward.

4.3.5 Deleting Text

The described commands for moving and entering text and for deleting individual characters suffice for elementary editing.

Two new deleting functions (in addition to <**DELETE**>) are described in this section:

<**DELETE**>	Deletes the character in front of the work point
<**CTRL-D**>	Deletes the character after the work point (delete)
<**CTRL-K**>	Deletes the rest of the line (kill line)

Assume that the buffer contains the following text:

```
Woozle
```

The work point is placed in front of the “z”. After the <**DELETE**> command is given, the result will be:

```
Wozle
```

If <**CTRL-D**> command is selected instead of <**DELETE**>, the result will be:

```
Woole
```

The <**CTRL-K**> command is used to delete an entire line. This command deletes the text located between the work point and the next line change. The “flag” (invisible code) for the line change itself remains. On the other hand, if the work point is located immediately before a line change, then the line change will be deleted (the next line will be linked with the previous line).

Assume that the buffer contains the text:

```
Woozle
Wizzle
```

The work point is placed in front of the “z” on the first line. After the command <**CTRL-K**> is given, the result will be:

```
Woo_
Wizzle
```

The work point is now located after the word “Woo”. If the command <**CTRL-K**> is given again, the result will be:

```
WooWizzle
```

The line change has been deleted.

When an entire line is to be removed, the work point is placed at the beginning of the line. <**CTRL-K**> is then pressed twice. The first time deletes the text forward to the line change. The second depression deletes the line change.

Additional functions for deleting, moving and copying sections of text are described further in Section 4.11 Erasing and Moving Text.

4.3.6 Segments

This section describes the manner in which an edited or newly-written text is written into a segment and saved.

The segment handling commands do not have their own keys. Instead, the X-character (for “extended”) is used, followed by the control character for the desired command.

`<CTRL-X> <CTRL-S>` `(Save)`

Writes the buffer content into the segment.

`<CTRL-X> <CTRL-V>` `(Visit)`

Writes the segment into the buffer.

The command <**CTRL-X**> <**CTRL-S**> ensures that the text in the buffer is written into the segment. This command is used before AMIS is escaped. It should also be used occasionally while editing in order to prevent the loss of results in case of a malfunction or a power outage.

The command <**CTRL-X**> <**CTRL-V**> causes the content of a specified segment to be copied and written into the work buffer. The command asks for the segment's name. After the name is given, press <**ENTER**>.

A new segment is created in precisely the same way. The new segment name is written in when the question "Visit segment" is displayed. The system then writes "New segment" on the echo line. When the text is ready to be saved, the command <**CTRL-X**> <**CTRL-S**> is used. With <**CTRL-X**> <**CTRL-V**> the user can move back and forth between different segments without leaving AMIS. The handling of segments is described in further detail in Section 4.17 Handling Segments and Buffers.

4.3.7 Escaping Difficult Situations

Extremely difficult situations can sometimes occur. Such situations can be caused by a misconception, or by depression of the wrong key.

One way to escape such situations is to press <**CTRL-G**> one or more times. AMIS will then terminate its current activity and return to interpreting commands in the usual manner.

The <**CTRL-G**> command does not cover all of the situations which might occur. A description of alternative methods which can be used is presented in Section 4.7 How to Escape.

4.3.8 Escaping AMIS

AMIS is escaped by using the following command:

```
<CTRL-X> <CTRL-Z>
```

The contents of the buffer should be saved in a segment before this command is given. In practice, the command for escaping AMIS should be considered as:

```
<CTRL-X> <CTRL-S> <CTRL-X> <CTRL-Z>
```

A more detailed description of various ways to escape AMIS is presented in Section 4.7 How to Escape.

4.3.9 Help Commands

There are many ways to get help in AMIS. Two possibilities are described in this section. Additional possibilities are described in Section 4.8 Help and Documentation Commands.

AMIS includes a special help character. It is <**CTRL-_**>.

<**CTRL-_**> provides different types of help, depending on the situation at hand. For this reason, <C**TRL-_**> can be used when unclear questions are presented, or if the system acts in a peculiar manner when a command is given. The help command often provides an explanation.

When <**CTRL-_**> is depressed, the system usually writes the following on the echo line:

```
Doc (? for help):
```

"Doc" stands for "document".

An additional character must be entered to specify the type of help needed. A question mark (?) or <**CTRL-_**> provides a list of alternatives.

The other help possibility provides information about how commands function. This help is obtained by pressing <**CTRL-_**> and specifying the character C (Character). In answer to the question “Character:”, the user can give the command to which the help pertains.

Example:

```
<CTRL-_>
Doc (? for help): <C>
Character: <CTRL-F>
Description of the command <CTRL-F>
```

The information is presented as an overwritten message.

4.3.10 Using Blank Lines

When new lines are to be inserted into a text, the insertion can be done in a number of ways. Assume that the following text is in the buffer:

```
Wizzle_
Wozzle
```

The text “Pooh” is to be inserted between these two lines.

In this case, the easiest thing to do is to first insert a blank line. This is done with the command <**CTRL-O**>. The text “Pooh” is then written in on the blank line:

```
Wizzle

Wozzle
```

<**CTRL-O**> inserts a line change after the work point. If the work point is placed at the beginning of the line, the <**CTRL-O**> command will therefore yield a blank line. The letter “0” stands for “Open Line”.

If the work point is placed in front of the word “Wozzle”, the <**CTRL-O**> command will yield the following result:

```
Wizzle

Woozle
```

The work point remains on the blank line. The text “Pooh” is written in.

```
Wizzle
Pooh_
Woozle
```

It is also possible to use <**CTRL-U**> <**CTRL-U**> <**CTRL-O**> to insert several blank lines. This procedure is described in further detail in Section 4.5 Numeric Arguments.

The command <**CTRL-X**> <**CTRL-O**> is used as a convenient way to delete unused blank lines. This command deletes the blank lines which surround the work point, but retains the line in which the work point is located.

4.4 Start of AMIS

This section describes startup and editing with control characters.

Syntax:

`AMIS` `[[`*`volume`*`]:`*`segment.extension`*`]`

4.4.1 Semantics

The AMIS command starts AMIS, thereby starting the editing of the segment specified. If the specified segment already exists, it will be read into the work buffer. Otherwise, the editor will start with an empty buffer.

If no segment name is given, AMIS will start with an empty work buffer. The new segment is created with the command <CTRL-X> <CTRL-V> (see Section 4.3.6 Segments).

If no volume name is given, the system will use the last volume name given.

Example:

```
*AMIS SRCE:SEGMENT.SA
```

Starts editing of the segment SRCE:SEGMENT.SA.

4.4.2 Internal Segments

When the editor starts, an internal segment is created. This segment is used to store the text during editing.

This internal segment is created on the same volume as the segment (to be edited) is located on.

When the editor is started without a volume name having been indicated, the system will use the last volume name specified. If no volume name was specified earlier, AMIS will ask for the name before editing can begin.

There must be space available on the volume for the internal segment. Allowance must also be made for the addition of more text, which will make the internal segment larger than the original segment.

The internal segment is deleted when AMIS is escaped. However, it can remain if the editor has been escaped by means of “abort”. There is no way to recreate a usable segment from this internal segment alone. The segment, which is called AMISWORK.ET, should therefore be deleted.

4.4.3 Backup Segments

While editing an existing segment, one normally keeps the original segment. The primary reason for this is to ensure that the original segment is protected in case a malfunction (such as a power outage) occurs during editing, or while the edited segment is being input. The original segment can also be useful when faulty editing has been carried out.

The original segment automatically receives the extension BP (backup) when the new edited segment is input. If the backup segment (original segment) is to be used instead of the edited segment, proceed as follows:

1. Delete the edited segment:

   ***DELETE volu:segname.ex**

2. Rename the backup segment:

   ***COPY volu:segname.BP>:*.ex**

However, there are commands which write the edited segment over the original segment. In such cases, the amount of space used up on the diskette is not as great. However, if such a procedure is unsuccessful, both the original segment and the edited segment may be lost.

The commands for handling segments are described in Section 4.17 Handling Segments and Buffers.

4.4.4 Editing with Function Keys

When the start command AMIS is used, it is possible to edit using the function keys. The EDIT command described in Section 3.2.1 Start Editing is thus a subset of AMIS. However, the EDIT command should be used if only function keys are going to be used during editing. This command eliminates the danger that some control character might be input by mistake, perhaps thereby distorting the text.

4.5 Numeric Arguments

It is possible to assign a numeric argument to each command. The function of the argument is determined by the command to which it is assigned.

The basic form for a numeric argument is the command <**CTRL-U**> followed by the digits which comprise the argument.

The command **<CTRL-U>** causes the command line to be displayed at the bottom of the screen.

Example <**CTRL-U**> <**1**><**7**> <**CTRL-F**> means command <**CTRL-F**>, argument 17.

Negative arguments are allowed. They are written with a minus sign preceding the digits.

If no numeric value is indicated, the default value is 4. This is extremely useful in certain instances, for example, when several blank lines are needed (<**CTRL-U**> <**CTRL-O**>), or when the work point is to be moved down several lines (<**CTRL-U**> <**CTRL-N**>). It is also convenient whenever commands which only recognize the existence of one argument are used.

The arguments for <**CTRL-U**> are multiplicative. This means that the argument in the command <**CTRL-U**> is multiplied by any arguments preceding <**CTRL-U**>.

Example:

<**`CTRL-U`**> <**`CTRL-U`**> has the argument value 16,

<**`CTRL-U`**> <**`CTRL-U`**> <**`CTRL-U`**> has argument value 64.

This function is useful in cases such as these:

`<CTRL-U> <CTRL-U> <CTRL-F>`

Moving 16 characters forward on the line.

`<CTRL-U> <CTRL-U> <CTRL-B>`

Moving 16 characters backward on the line.

`<CTRL-U> <CTRL-U> <CTRL-N>`

Moving downward 16 lines.

`<CTRL-U> <CTRL-U> <CTRL-U> <CTRL-O>`

Inserting 64 blank lines.

Command repetition is one example of an argument function. The command <**CTRL-F**> (go forward one character) combined with the argument 17 will perform the function of moving forward 17 characters.

Another example group consists of commands which recognize the existence of the argument, but not its value. The command <**ESC**><**Q**> is used to rearrange sections of text so that the lines are filled out. If an argument is assigned to this command, the command will then also set a justified right-hand margin. The argument’s value is meaningless.

Yet another example group includes commands which have a special meaning for the argument. One such command is <**CTRL-K**>. Without an argument, this command must be given twice in order to delete a written line and the blank line left behind. With the argument, the command deletes as many lines as the argument indicates. Another such command is <**CTRL-V**>. Without an argument, this command moves the window forward one screen page in the buffer. With an argument, the window is moved as many lines as the argument indicates.

4.6 How Commands are Written

There are several different types of command in AMIS. The use of CTRL- and ESC-(META-) commands is described in Section 4.3.3 Writing Commands. This section concerns the principles involved in the various commands.

4.6.1 Expansion to More Commands

Combinations of <**CTRL-**>, <**ESC**> and <**CTRL**><**ESC**> provide a large number of commands. However, these combinations do not suffice to meet the requirements of AMIS. For this reason, AMIS also uses multi-character commands.

The multi-character commands are preceded by <**CTRL-X**> or <**ESC**><**X**>. The “X” stands for “extend” (X-tend). <**CTRL-X**> must be followed by additional characters. <**ESC**><**X**> presents the question “Command:” at the bottom of the screen (described further in the next section).

The multi-character commands which are preceded by <**CTRL-X**> are used less often. Examples of such commands include the segment handing commands <**CTRL-X**> <**CTRL-V**> and <**CTRL-X**> <**CTRL-F**>. Each group of commands has similar characteristics. This makes them easier to remember. The <**CTRL**> commands are used most frequently. The <**ESC**> commands are often more powerful variants of corresponding <**CTRL**> commands.

An example of this would be:

`<CTRL-F>`

Moves one character forward.

`<ESC><F>`

Moves one word forward.

4.6.2 Named Commands

All commands have names. The names describe succinctly what the commands are used for. They usually consists of two words. A typical example is “Replace String”.

The commands described earlier, which use the letter combinations <**CTRL-**>, <**ESC**>-, <**CTRL**><**ESC**>- and <**CTRL-X**> also have names, so that they can be differentiated. However, these names are used very rarely and are therefore of little significance.

On the other hand, the commands which are introduced with <**ESC**><**X**> require names. The command’s name is given as the response to the system’s question after the letter combination.

A command from this group is introduced by <**ESC**><**X**>.

The system will ask:

```
Command:
```

The name of the command is entered and concluded with <**ENTER**>.

One is allowed to abbreviate the command names word for word, as long as they remain unambiguous. When the words are abbreviated, the system fills in the omitted characters when the space key is struck.

If <**ESC**> is struck instead of the space key, AMIS will attempt to fill in the entire command name.

If a question mark (?) is written in this position, all of the command names which are conceivable are listed in an overwritten message.

Commands which are not normally written as named commands have names which start with the letter “R”.

A named command can be assigned one or more text arguments. The arguments are written either directly following the command name or as responses to AMIS questions.

If the argument is to be written directly following the command name, this should be done before <**ENTER**> is struck. The command name and the argument are separated by the character for ESCAPE ($). This means that <**ESC**> must be struck after the command name and between arguments.

The character for ESCAPE is written as $ in both the documentation and on the echo line.

If the arguments are omitted after the command name, AMIS will ask about the necessary arguments. These questions are displayed on the echo line.

Replace String is an example of a named command which requires text arguments. This command is used to replace one text with another. If this command is given as a named command with the arguments written directly, it has the following appearance:

```
<ESC-X> Replace String$woozle$wizzle
```

The $-character separates the command and both arguments. This character symbolizes ESCAPE. If the arguments are omitted when the command is given, it will have the following appearance:

```
<ESC-X> Replace String
```

This command will then cause AMIS to ask its first question:

```
Replace:
```

The answer, which corresponds to the word which is to be replaced, is “Woozle” in this particular case. This word is entered and concluded with <**ENTER**>. This causes AMIS to ask another question:

```
Replace woozle with:
```

The word chosen to replace “wozzle” is “wizzle”. This replacement word is entered and concluded with <**ENTER**>.

4.7 How to Escape

This section describes the possibilities which exist for escaping certain positions. The section describes how to escape AMIS, and how to get out of certain situations.

4.7.1 Escaping AMIS with <CTRL-X> <CTRL-Z>

AMIS is escaped with the command:

```
<CTRL-X> <CTRL-Z>
```

The contents of the buffer should be saved in a segment before this command is given. In practice, the command for escaping AMIS should be considered as:

```
<CTRL-X> <CTRL-S> <CTRL-X> <CTRL-Z>
```

or

```
<F1> <CTRL-X> <CTRL-Z>
```

The command <**CTRL-Z**> <**Z**> can also be used to escape AMIS, or to escape whatever level of AMIS one may be in. If <**CTRL-X**> <**CTRL-F**> has been used for moving and editing in different segments, all of these buffers will be listed in an overwritten message. The buffers which are not to be saved are marked with an asterisk (*).

If the command <**CTRL-X**> <**CTRL-S**> is not given to save an edited buffer when AMIS is about to be escaped, the system will display a question and a list on the screen:

```
Modified buffers exist. Quit? (Y or N)?
```

The answer “N” will return the system to editing mode, from which the buffer can be saved using the command <**CTRL-X**> <**CTRL-S**>.

If the answer “Y” is given, AMIS will be escaped and the contents of the buffer will not be saved.

4.7.2 Interruption With <CTRL-G>

An initiated command can be interrupted by the command <CTRL-G>. This can be used if, for example, ESCAPE or a search command is given by mistake.

If an incorrect command has been executed, for example, if text has been deleted, it is sometimes possible to undo the result. This must be done immediately following the incorrect command. The command for this is:

```
<ESC> <X> Undo
```

4.7.3 Reading the Mode Line

If the mode line is enclosed in parentheses, for example, (Query Replace), it means that a command is being executed. If help is needed, use <**CTRL**>. The command can be interrupted by <**CTRL-G**>.

If the mode line is enclosed in brackets, for example, [Kbd Query.], it means that a command has called AMIS recursively. This is described in the next section.

4.7.4 Recursive Calling of AMIS

There are some commands which make it possible to temporarily edit a text other than the one which is presently being worked on. An example of such a command is Query Replace. This command temporarily interrupts a search in order to perform other editing. This procedure is called recursive editing, and it is described in Section 4.13 Replacing Text.

A recursive edit can be escaped in two different ways. The normal command for this purpose is <**CTRL-Z**> <**Z**>. This is the same command used to escape AMIS. Another alternative is <**CTRL**><**]**>. This command interrupts the command from which the recursive editing was done. The command <**CTRL-G**> cannot be used to escape a recursive edit.

If the command <**CTRL-X**> <**CTRL-Z**> is used in a recursive edit, the recursive edit will be escaped. In addition, a switch to the command level will also occur.

4.8 Help and Documentation Commands

The use of the help character <**CTRL-_**> is described in Section 4.3.9 Help Commands. In that section, it was indicated that another character was needed in order to define the type of help required. AMIS asks for that other character by writing “Doc (? for help):” on the echo line (“Doc” stands for “document”).

This section concerns the most important alternatives.

4.8.1 Command Description from Character (C)

The letter “C” (Character) after the help character produces a description of a command, which is specified by control character.

Example:

```
<CTRL-_>
Doc (? for help):<C>
Character: <CTRL-D>
Description of the command <CTRL-D>
```

4.8.2 Command/Variable Description from Name (D)

The letter “D” (Describe) after the help character produces a description of a command or a variable, which are specified by name.

Example:

```
<CTRL-_>
Doc (? for help):<D>
Describe: EXIT
Description of the command EXIT
```

4.8.3 Command Character from Name (W)

The letter “W” (Where is) after the help character produces the character combination for a command, which is specified by name.

Example:

```
<CTRL-_>
Doc (? for help):<W>
Where is: DELETE
CTRL-D>
```

4.8.4 Command Name from Text String (A)

The letter “A” (Apropos) after the help character is used to search for a command name. This function lists all of the commands which have a given text string in their names.

Example:

```
<CTRL-_>
Doc (? for help):<A>
Apropos: NEXT
List of all commands containing the text string "NEXT".
```

The text string must be selected in such a way that it reflects the function being sought.

Suitable search words for text strings are:

> Character, line, word, sentence, paragraph, region, page, buffer, screen, window, segment, beginning, end, case, mode, forward, backward, next, previous, up, down, search, kill, delete, mark, fill, indent.

It is sometimes necessary to conduct searches a number of times, using different search words, before the sought command is located.

4.8.5 Information Regarding Help Character (<CTRL-_>)

An additional help character <**CTRL-_**> after the initial help character provides information on the available alternatives to the help character.

Example:

```
<CTRL-_>
Doc (? for help):<CTRL-_>
```

Information on the available alternatives to the help character.

4.9 Moving the Work Point

This section deals with the commands used to move the work point in the buffer. In addition to the commands described in this section, it is also possible to use the search commands described in Section 4.12 Search.

4.9.1 Moving Short Distances

The following commands are used to move the work point forward and backward on a line. Movement over line changes means that the work point will be moved to adjacent lines.

<**CTRL-F**>	Moving forward one character
<**CTRL-B**>	Moving backward one character
<**ESC**> <**F**>	Moving forward one word
<**ESC**> <**B**>	Moving backward one word

The following commands are used to move to other lines, and to move to the beginning or the end of a line.

<**CTRL-P**>	Moving up one line
<**CTRL-N**>	Moving down one line
<**CTRL-A**>	Moving to the beginning of a line
<**CTRL-E**>	Moving to the end of a line
<**ESC**> <**ENTER**>	Moving to the beginning of a line (after initial blank characters).

The commands <**CTRL-N**> and <**CTRL-P**> move the work point up or down, without moving forward or backward along the line. However, this assumes that either written or blank characters are present in the intended positions. Otherwise, the work point will be moved to the last character in the line.

4.9.2 Moving Relative to Sentences and Characters

The following commands are used to move between sentences and paragraphs:

<**ESC**> <**A**>	Moving backward one sentence
<**ESC**> <**E**>	Moving forward one sentence
<**ESC**> <**[**>	Moving to the beginning of a paragraph
<**ESC**> <**]**>	Moving to the end of a paragraph

These commands assume that the sentences and paragraphs are constructed in the traditional manner. This means that a sentence ends with a punctuation mark, followed by at least one blank character or a line change. A paragraph is assumed to be ended if it is followed by a blank line, or if the next sentence does not begin in the first position on the next line. Only empty lines are considered as criteria for the ends of paragraphs in connection with the editing of program text.

4.9.3 Moving Longer Distances

Several commands can be used to move the work point longer distances in the buffer. These commands are presented in this section:

<**CTRL-V**>	Moving forward one screen page
<**ESC**> <**V**>	Moving backward one screen page
<**CTRL-X**> <**[**>	Moving to the beginning of a page
<**CTRL-X**> <**]**>	Moving to the end of a page
<**ESC**> <**<**>	Moving to the beginning of the buffer
<**ESC**> <**>**>	Moving to the end of the buffer

The command pair <**CTRL-V**> and <**ESC**> <**V**> allows the user to leaf through the text screen by screen.

“ L” is used to change pages. This character can be inserted by using the command <**CTRL-Q**> <**CTRL-L**>.

When working with large segments, it is often easier to use the search commands to move the work point to a given location.

4.10 Mark and Region

Many commands operate within a region in the buffer (not just at the work point). Starting and stopping positions must be specified for such operations. One of these positions will be indicated by the work point, while the other will be indicated by the “mark”. The text located between the mark and the work point comprises the “region”. Either the mark or the work point can be used to indicate the starting position (with respect to the stopping position).

4.10.1 Moving the Mark

The mark works like another work point. The mark has no symbol and therefore cannot be seen. For this reason, it is advisable to place the mark immediately before using it, so that the user has the placement memorized.

The position of the mark is set by moving the work point to the desired position, and then giving the command <.> . The positions of the mark and the work point can be exchanged by giving the command <**CTRL-X**> <**CTRL-X**>.

The following commands apply to the mark:

<.>	Places the mark at the work point
<**ESC-@**>	Moves the mark forward one word
<**CTRL-X**> <**CTRL-X**>	Switches the positions of the mark and the work point

4.10.2 Setting the Region

A region is set in the following way.

1. Place the work point at one of the desired region’s limit points.
2. Give the command <.>.
3. The mark is now set.
4. Move the work point to the region’s other limit point.
5. The region is now set.

Check the region by giving the command <**CTRL-X**> <**CTRL-X>**. The work point will alternate between the starting and stopping positions each time the command is given.

There are several commands which direct the mark to a region. These commands place both the mark and the work point. These commands are:

<**ESC**> <**H**>	Sets the region around a paragraph
<**CTRL-X**> <**CTRL-P**>	Sets the region around a page
<**CTRL-X>** <**H**>	Sets the region around the entire buffer

4.10.3 Other Uses of the Mark

There are commands which move the work point far away from its initial position. Some of these commands automatically set the mark at the work point’s initial position so that it can be found again easily. Typical examples of such commands are the search commands <**CTRL-S**> and <**CTRL-R**>. These commands set the mark at the work point’s initial position if the search procedure moves the work point more than 500 characters.

One type of command sets a region around an edited section of text. This is true of commands which insert sections of text when, for example, they are being moved or copied from another segment. This function makes it very easy to continue work on the text section in question.

4.11 Erasing and Moving Text

This section describes the commands used to erase and move text. These functions are described together because they make use of the same commands, to a certain extent.

4.11.1 Erasure

Two types of erase commands are found in AMIS. One type includes those commands which are used to remove individual characters or blank lines. The other type consists of those commands which remove entire words, lines, sentences or regions. The first type of commands are called Delete commands, while the second type are called Kill commands.

Text which is removed by Delete commands is lost entirely. On the other hand, text removed by Kill commands is saved in a special erase buffer. This buffer can then be used as a normal text segment, i.e., it can be copied and inserted wherever one wishes. Using the erase buffer provides a convenient means of moving and copying text.

4.11.2 Delete

These commands remove characters without saving them:

<**CTRL-D**>	Removes the character to the right of the work point
<**DEL**>	Removes the character to the left of the work point
<**ESC**> <\>	Removes blank and tab characters around the work point
<**CTRL-X**> <**CTRL-0**>	Removes empty lines from around the work point line
<**ESC**> <^>	Links the line to the preceding line

4.11.3 Kill

These commands remove text, and they also save the text in the erase buffer. If several Kill command operations are carried out in sequence, the combined text which has been removed will be saved.

<**CTRL-K**>	Removes the rest of the line
<**CTRL-W**>	Removes a region
<**ESC**> <**D**>	Removes the word after the work point
<**ESC**> <**DEL**>	Removes the word in front of the work point
<**CTRL-X**> <**DEL**>	Removes text from the beginning of the sentence
<**ESC**> <**K**>	Removes the rest of the sentence

An entire line can be removed by placing the work point at the start of the line and then giving the command <**CTRL-K**>. The line will become empty up to the line change. Giving the command <**CTRL-K**> will remove the remaining line change.

If several lines are to be removed, the command <**CTRL-K**> can be repeated. Another alternative is to assign the command <**CTRL-K**> an argument. The argument indicates the number of lines to be removed.

A region is removed by first specifying its extent by using the mark and the work point, and by then giving the command <**CTRL-W**>.

If one or several isolated words are to be removed, it is easiest to use <**ESC**> <**D**> or <**ESC**> <**DEL**>. An incorrectly written word is most easily removed with <**ESC**> <**DEL**>. Similarly, <**CTRL-X**> is the easiest way to remove an incorrectly written sentence start.

4.11.4 Moving and Copying Text

If a text is to be moved (to one or more locations), it is first placed in the erase buffer through the use of a Kill command. The work point is then moved to the desired text location. The command <**CTRL-Y**> inserts the text into the desired location. Repeating the command <**CTRL-Y**> duplicates the text in the locations selected. In this manner, words, lines, sentences regions can be moved and duplicated.

<**CTRL-Y**>	Inserts last removed text at work point.
<**ESC**> <**Y**>	Replaces removed text with older removed text.
<**ESC**> <**W**>	Copies the region and saves it in the erase buffer.
<**CTRL**> <**ESC**> **<W>**	Text from the next kill command is added to the erase buffer.

Copying is carried out in the same way as moving text, with the exception that the original text is left in place. The command <**ESC**> <**W**> is used for copying. This command copies a marked region and writes the text into the erase buffer. It therefore functions in the same way as the command <**CTRL-W**> followed by <**CTRL-Y**>.

The command <**CTRL-Y**> is used to insert text from the erase buffer into one or more locations.

AMIS saves the 8 most recently removed texts in the erase buffer. The command <**CTRL-Y**> is used to insert the last removed text into a selected location. If a previously removed text is desired, the command <**ESC**> <**Y**> is used to switch from the last removed text to the next to last removed text. Repeated use of <**ESC**> <**Y**> will move one back to older texts, one increment at a time. In this way, the last 8 removed texts can be reused.

<**ESC**> <**Y**> is used only in this way, after the command <**CTRL-Y**>.

4.11.5 Q-registers

Use of the erase buffer entails a temporary storage of the removed text. More permanent storage is available through the use of the Q-registers. There are many Q-registers, with various letter names.

<CTRL-X> **<***X***>** *q*	Copies the region to the Q-register q.
<CTRL-X> **<***G***>** *q*	Inserts text from the Q-register q.

“q” stands for the name of the desired Q-register, which is specified by a letter A-Z.

<**CTRL-X**> <**X**> ***q*** means that a copy of the area specified by the mark and the work point will be made. The copy is input into the selected Q-register.

<**CTRL-X**> <**G**> ***q*** means that the text in the indicated Q-register will be inserted at the position where the work point is located.

4.11.6 Switch-Places Commands

Switching places of two items are done by the following commands:

<**CTRL-T**>	Two characters switch places.
<**ESC**> <**T**>	Two words switch places.
<**CTRL-X**> <**CTRL-T**>	Two lines switch places.
<**CTRL-X**> <**T**>	Two regions switch places.

When switching characters, the work point is placed after the last character. Then the command <**CTRL-T**> is given.

Example:

```
SE_

<CTRL-T>
ES_
```

When switching words, the work point is placed between the words which are to switch places. Then the command<**ESC**> <**T**> is given.

Example:

```
AND _ OR

<ESC> <T>
OR AND_
```

When switching lines, the work point is placed on the lower of the two lines to be switched. The command <**CTRL-X**> <**CTRL-T**> is then given.

Example:

```
AND WIZZLE
OR WOZZLE

<CTRL-X> <CTRL-T>

OR WOZZLE
AND WIZZLE
_
```

After the switch, AMIS places the work point in front of the next character, word or line. Repetitions of the command will cause the character, word or line to move through the text incrementally.

The command <**CTRL-X**> <**T**> is used to switch the contents of two specified regions. The two regions are specified in the following manner:

- One region is specified using the mark and the work point in the usual fashion.
- The other region consists of the text between the mark's previous position and its position prior to that previous position.

Example:

Assume that the contents of region X, with the borders a and b, are to be switched with the contents of region Y, which has the borders c and d. The procedure would be:

- Set the mark at a
- Set the mark at b
- Set the mark at c
- Place the work point at d
- Give the command <**CTRL-X**> <**T**>

4.12 Search

As the text increases in size, it becomes more difficult to find things within it. AMIS has commands for searching both forward and backward, thereby facilitating search procedures.

4.12.1 Incremental Search

To do an incremental search use the following commands:

<**CTRL-S**>	Incremental search forward
<**CTRL-R**>	Incremental search backward

The command <**CTRL-S**> starts an incremental forward search process. The command <**CTRL-R**> starts a corresponding backward search process.

Once the search process has begun, it is possible to write in the text which is being searched for. It is also possible to give subcommands to control the search.

The work point's prospective placement is presented in the window at the same time the search text is being written in and the subcommands are given.

The search process is terminated by using **ESCAPE**. The work point will then move to its new location.

This search process is called an "incremental search".

The text "I-Search:" or "Reverse I-Search:" is placed on the echo line while the search is in progress. The search text is displayed after the colon. Text which is being inserted is harbored in this text string. The work point moves forward to the first occurrence of the corresponding text string. The characters in the text string are not echoed directly, but rather presented on the echo line in the order in which the editor finds the various characters.

No differentiation is made between upper- and lower-case letters in the search process.

4.12.2 Subcommands

The following commands can be given during a search:

<**CTRL-S**>	Searches forward for next occurrence
<**CTRL-R**>	Searches backward for next occurrence
<**DEL**>	Deletes last character written in search string
<**ESC**>	Terminates search
<**CTRL-G**>	Terminates search. Returns to initial position

If a search proves unsuccessful, the search string can be expanded. Alternatively, <**CTRL-S**> or <**CTRL-R**> can be pressed repeatedly to search for the next occurrence of the same string.

If the search string specified last is to be used again, it can be recalled by pressing <**CTRL-S**> or <**CTRL-R**>. This applies only when no search string has been written in.

If the search moves the work point more than 500 characters, the mark will automatically be set to the starting position. In such cases, "*** Mark Set ***" will be indicated on the echo line. The mark is used to facilitate return to the starting position.

4.12.3 If a String is not Found

If AMIS does not find the correct string, "Failing I-Search:" or "Failing Reverse I-Search:" will be displayed on the echo line. If the search is to be concluded at this point, this is done by pressing <**ESC**>.

An unsuccessful search may be due to an incorrectly written search string. Deletion is carried out character by character, using <**DEL**>. If a forward search is unsuccessful, it may be due to the fact that the section being sought is located back in the text. Continue the search with <**CTRL-R**>.

If a search is terminated with <**CTRL-G**> will be returned to the pre-search positions. The work point will be moved back to the starting position.

4.12.4 Single Search

If the ESCAPE command is given directly after <**CTRL-S**>, a search is conducted until the first occurrence is found, after which the search is concluded. This type of search may be preferable in certain instances.

The following commands are used in single searching:

<**ESC**>	Searches for string and then terminates search.
<**CTRL-R**>	Changes the search direction.
<**CTRL-B**> <**CTRL-S**>	Searches forward from the beginning of the segment.
<**CTRL-E**> <**CTRL-S**>	Searches backward from the end of the segment.
<**CTRL-F**> <**CTRL-S**>	The text section being sought is displayed at the top of the screen.
<**DEL**>	Deletes the last character written in the text string.
<**CTRL-S**>	Searches for the next occurrence.
<**CTRL-G**>	Terminates search. Returns to initial position.

4.13 Replacing Text

It is sometimes necessary to replace one text string with another throughout the text, or just at certain points. In either case, the command for selective replacement can be used:

<**ESC**> <**%**>	Selective replacement.

This command causes two questions to be asked. The first question asks which string is to be deleted. The second question asks which string is to replace the deleted string.

After these questions have been answered, AMIS will perform a search from the work point to the end of the buffer. When a string which corresponds to the string which is to be deleted is found, the work point moves to it. At this point, AMIS will wait for one of the following responses:

<**SPACE**>	Replace the text string. Search further.
<**DEL**>	Do not replace the text string. Search further.
<**ESC**>	Terminate the activity.
<**.**>	Replace the text string, then terminate the activity.
<**!**>	Replace the text string here and in the rest of the buffer.
<**,**>	Replace the text string. Search no further.
<**CTRL-R**>	Recursive editing. Return with <**CTRL-Z**> <**Z**>.
<**CTRL-W**>	Remove string. Recursive editing. Return with <**CTRL-Z**> <**Z**>.

If replacement is to occur throughout the entire buffer, the work point is placed at the beginning of the buffer. This is done before the selective replacement begins. While the activity is going on, <**SPACE**> or <**DEL**> is pressed to replace or skip over each string.

If the replacement is to occur throughout, an exclamation mark is used as the response.

No differentiation is made between upper- and lower-case letters in the search process.

Upper- and lower-case letters for the replacement string are determined as this string is written in. This disposition is retained upon replacement, since it agrees with the letter disposition of the replaced string.

4.14 Major Modes and Minor Modes

AMIS can easily be adjusted for different types of text. Such adjustments are achieved through the use of a major mode and a minor mode. The major mode indicates the type of editing to be performed. The minor mode indicates various editing functions which can be selected, regardless of the major mode.

4.14.1 Major Modes

There are three major modes in AMIS. The applications of the various major modes are presented in the following three sections. Switching amongst the various major modes is performed using the following named commands:

<**ESC**> <**X**>	Fundamental Mode	Basic Mode
<**ESC**> <**X**>	Text Mode	Text Mode
<**ESC**> <**X**>	Keypad Mode	Function Key Mode

The system's mode is indicated by the first word in parentheses on the mode line.

Information regarding the characteristics of the various modes is obtained by giving the command <**CTRL-_**> <**D**> (Describe).

Example:

```
<CTRL-_> <D> Text Mode
```

4.14.1.1 Fundamental Mode

The fundamental mode is a basic mode which offers no adjustment functions for special types of editing.

4.14.1.2 Text Mode

The text mode is used to edit all text which does not comprise program text.

Indented lines are perceived as the beginnings of new paragraphs. The tab key is used to place the work point at specified tab stops. These stops are permanently placed in positions 9, 17, 25, etc.

Section 4.15 Special Text Commands describes some of the functions which are available in text mode.

4.14.1.3 Keypad Mode

Only the function keys can be used in this mode. If a control character is entered, the error message "Illegal command" is presented.

If AMIS is started by the EDIT command, the keypad mode will be selected automatically. If another major mode is selected later, the control character commands can be used once again.

Switching to another major mode is achieved by using <**O**> or <**ESC**> <**X**> (on keypad), at which point the question "Command" will be displayed. The user responds with the desired major mode.

4.14.2 Minor Modes

There are two minor modes in AMIS. The minor modes are used to select or reject specified functions, regardless of the major mode. Descriptions of the various minor modes are presented in the following two sections.

Switching amongst the various minor modes is performed using the following named commands:

<**ESC**> <**X**>	Overwrite Mode	Overwriting
<**ESC**> <**X**>	Auto Fill Mode	Automatic line change

All of these minor modes are available simultaneously. For example, the mode line might read:

```
AMIS (Fundamental Fill Swedish Overwrite)
```

In this situation, if the user presses <**ESC**> <**X**> and then writes “Swedish” on the command line, the minor mode “Swedish Mode” will be de-selected. The mode line will then be changed to:

```
AMIS (Fundamental Fill Overwrite)
```

The other minor modes can be de-selected in the same fashion. The modes which are in force at a given time are displayed on the mode line, in parentheses, following the major mode.

4.14.2.1 Overwriting

In the overwriting mode, the characters are not inserted at the work point, but written over the existing characters.

This function can be used when, for example, a new text is to be fitted in over the same space as a previous text. If this mode is not reset, the new text will be written over text which is to be saved. Therefore, de-select this mode immediately after each use.

4.14.2.2 Automatic Line Change

In automatic line change mode, inserted running text will automatically be broken into lines of a specific length. The line length is determined by the margin settings (Section 4.15.3 Margin Settings).

4.15 Special Text Commands

This section concerns commands which are particularly useful in working with running text.

When editing Swedish text, the major and minor modes selected should be “Text Mode” and Swedish Text”, respectively.

The commands which were described earlier, and which are used to handle words, sentences and paragraphs, are presented in the following list. Several new commands are then described.

<**ESC**> <**F**>	Moving forward one word.
<ESC> <**B**>	Moving backward one word.
<**ESC**> <**D**>	Removes the word after the work point.
<**ESC**> <**DEL**>	Removes the word before the work point.
<**ESC**> <**@**>	Sets the mark forward one word.
<**ESC**> <**T**>	Two words switch places.

<**ESC**> <**A**>	Moving backward one sentence.
<**ESC**> <**E**>	Moving forward one sentence.
<**ESC**> <**K**>	Removes the rest of the sentence.
<**CTRL-X**> <**DEL**>	Removes text, starting from the beginning of the sentence.
<**ESC**> <**[**>	Moving to the beginning of a paragraph.
<**ESC**> <**]**>	Moving to the end of a paragraph.
<**ESC**> <**H**>	Sets the region around a paragraph.

4.15.1 Switching Between Upper- and Lower- Case Letters

The following commands provide convenient ways to switch between upper-and lower-case letters.

<**ESC**> <**U**>	Changes the next word to upper-case.
<**ESC**> <**L**>	Changes the next word to lower-case.
<**ESC**> <**C**>	Changes the first letter of the next word to upper-case.

After the command has been given, the work point is placed after the word to be changed. Repetitions of the command cause the change to move through the text word by word.

Several words can be changed simultaneously if a numeric argument is assigned to the command. Argument -1 will mean that the previous word is changed without moving the work point. This argument is extremely useful in case of a mistyped letter, or if the user forgets to hit the shift key.

In addition to the above-mentioned commands, there are two other commands which change all of the letters in a region.

<**CTRL-X**> <**CTRL-U**>	Changes all the letters in the region to upper-case.
<**CTRL-X**> <**CTRL-L**>	Changes all the letters in the region to lower-case.

4.15.2 Indented Text

Indentation is performed using the tab command <**CTRL-I**>. Additional commands for handling indented text are presented in this section.

<**CTRL-I**>	Indentation of a line.
<**CTRL-J**>	Means <CR> followed by <CTRL>-I.
<**ESC**> <^>	Links the indented line to the preceding line.
<**CTRL-X**> <**CTRL-I**>	Moves the text in a region sideways.
<**CTRL-ESC-**>	Indentation of an entire region.

The tab command <**CTRL-I**> must be given at the beginning of the line which is to be indented.

The effect of the tab command <**CTRL-I**> depends on the major mode. In “Fundamental Mode” and “Text Mode”, this command is used to move to the next tab stop (these stops are permanently set at 9, 17, 25, etc.). In “Program Mode”, this command will bring about an indentation which is adapted to the program text.

The easiest way to handle several indented lines in sequence is to use the <**CTRL-J**> command.

The <**CTRL-X**> <**CTRL-I**> command moves an entire region sideways. A numeric argument is assigned to the command to determine the number of positions involved in the movement. A positive argument will move the region to the right, while a negative argument moves the region to the left.

Example:

The command <**CTRL-U**> <**6**> <**CTRL-X**> <C**TRL-I**> yields the following display on the mode line:

```
C-U 6 ^X Tab
```

The region on which the command is used will be displaced 6 positions to the right.

4.15.3 Margin Settings

The text margins are handled by means of the following commands:

<**CTRL-X**> <**F**>	Sets the right margin.
<**CTRL-X**> <**.**>	Sets the left margin string.
<**ESC**> <**Q**>	Applies margins to a paragraph.
<**ESC**> <**G**>	Applies margins to a region.

The right margin is set by giving the command <**CTRL-X**> <**F**> (Fill Column) when the work point is located at the desired right margin column. This margin can also be set by assigning the command a numeric argument.

Automatic line changes with set margins are derived through the use of the “Automatic Line Change” minor mode (Section 4.14.2.2 Automatic Line Change).

If a change is to be made in the text or in a margin setting, the valid margins can be applied to a paragraph by using the command <**ESC**> <**Q**>. This margin application will be terminated at the end of the paragraph, i.e., at a blank line or a line which begins with one or more blank characters. If the margins are to be applied to several paragraphs, the command <**ESC**> <**G**> can be used for a selected region.

The left margin can be indented by preceding each line with an empty margin string. The length of this string will determine the indentation. The margin string is set by giving the command <**CTRL-X**> (Fill Prefix) when the work point is located at the desired left margin column. Once the left margin string is set, each line is automatically preceded by an empty margin string. Lines which are preceded by margin strings belong to the paragraph, i.e., they do not terminate a paragraph from a handling standpoint.

4.16 Multiple Buffers

The descriptions in the foregoing sections have only concerned one text in one buffer.

However, AMIS allows for several texts in various buffers simultaneously. This feature can be used to move text sections between different segments, or to edit two segments simultaneously, or, for example, to edit one segment while simultaneously obtaining editing instructions from another segment.

The segments' contents are written into the buffers in the manner described previously. After a segment is written into each buffer, the user can switch between them quickly and easily. It is not necessary to save the contents of a buffer when switching to another buffer. The contents will remain intact. On the other hand, all edited buffers must be saved in segments before AMIS is escaped.

The following commands are used to handle multiple buffers:

<**CTRL-X**> <**CTRL-B**>	Lists defined buffers.
<**CTRL-X**> <**B**>	Switches to another buffer (or creates a new one).
<**CTRL-X**> <**CTRL-F**>	Searches for (or creates) buffer with specified segment.

All buffers have names. The name of the first buffer is set by AMIS. It is called "Main". The other buffers are named by the user.

When the buffers are listed by the command <**CTRL-X**> <**CTRL-B**>, the buffers' names and the segments which they contain are presented. When the command <**CTRL-X**> <**B**> is used to switch to another buffer, AMIS asks for the name of the desired buffer. The switch over to the indicated buffer then takes place. However, if a name is specified which has not been defined for a buffer, a new buffer is created with that name.

The command <**CTRL-X**> <**CTRL-F**> is used both to create a new buffer and to write a segment into it. The name of the new buffer will automatically be the same as the segment's principal name. This command can be used to help the user switch easily from one segment to another.

It is possible to display the contents of two different buffers on the screen simultaneously. This can be accomplished by using two windows (described in Section 4.18 Two Windows).

4.17 Handling Segments and Buffers

The previous sections in this description have touched on segment handling superficially. A more detailed study of segment handling is presented in this section. The handling of buffers will also be considered to some extent.

Table 4-1. Summary of Segment/Buffer Handling Functions

Buffer handling:	
<**CTRL-X**> <**CTRL-F**>	Searches for buffer containing specified segment
<**ESC**> <**~**>	Removes change marker
Copying and moving from segment to buffer:	
<**CTRL-X**> <**CTRL-V**>	Segment's contents written into buffer
<**ESC**> <**X**> (Insert file)	Segment's contents inserted into buffer
Copying and moving from buffer to segment:	
<**CTRL-X**> <**CTRL-S**>	Buffer's contents written into opened segment
<**CTRL-X**> <**CTRL-W**>	Buffer's contents written into optional segment
<**ESC**> <**X**> (Write Region)	One region written into opened segment
No backup segment saved for specified commands:	
<**ESC**> <**X**> (No Backup Save File)	Applies to <**CTRL-X**> <**CTRL-S**>
<**ESC**> <**X**> (No Backup Write File	Applies to <**CTRL-X**> <**CTRL-W**>
<**ESC**> <**X**> (View File ...)	Contents of given segment will be listed on screen

<**CTRL-X**> <**CTRL-F**>	is used to search for a buffer which contains a specified segment. This command also provides a simple means for switching between buffers. This Command can also be used to write the segment's contents into an existing buffer, or a new one.
<**ESC**> <**~**>	Each buffer whose contents undergo some change is provided with a "flag". AMIS can thereby indicate to the user that a given buffer has been changed, in some cases. The command <**ESC**> <~> removes this "flag".
<**CTRL-X**> <**CTRL-V**>	causes the contents of a specified segment to be written into an empty buffer. This command asks the question: `Visit segment:` The desired segment name is given. If the volume name is omitted, the system will use the last name specified
<**ESC**> <**X**> (Insert file)	causes the text in a segment to be copied and inserted into the work buffer. The text will be inserted at the location of the work point. This command asks the questions "Insert segment:", which is answered in the same manner as in the case of <**CTRL-X**> <**CTRL-V**>.
<**CTRL-X**> <**CTRL-S**>	is used to write the contents of the buffer into a segment which is already opened. The name of the opened segment is displayed on the mode line. This command is used after editing has been concluded, as well as occasionally during the edit, to ensure that the editing which has been done will not be lost.
<**CTRL-X**> <**CTRL-W**>	is used instead of <**CTRL-X**> <**CTRL-S**> if the contents of the buffer are not to be written into the opened segment, but rather into another segment. This command asks the question: Write segment: The name of the segment into which the contents are to be written is given in response.
<**ESC**> <**X**> (Write Region)	is used to write a region into the segment which has already been opened. The handling procedure is the same as for <**CTRL-X**> <**CTRL-S**>, except that in this case the region must be specified by the mark and the work point before the command is given.
<**ESC**> <**X**> (No Backup Save File), <**ESC**> <**X**> (No Backup Write File) and <**ESC**> <**X**> (No Backup Write Region)	These commands are used when the original segment is not to be saved. The original segment is normally saved during editing. The edited text is placed in a new segment with the same identifier as before. The original segment retains the same name, but the extension BP (backup) is added.
<**ESC**> <**X**> (<0> View File)	This command lists the contents of a given segment on screen. Press <SPACE> to initiate scrolling to next page. Any other command will terminate displaying the contents on screen.

When these commands are used, the edited text is written "over" the original text in the original segment. For example, these commands can be used when the available space on the volume does not allow for both the original and edited segments.

If a volume has insufficient space to hold a given segment, an error message is presented. However, the system remains in AMIS. In such a situation, the system will delete the "partial segment". The original segment remains. On the other hand, if the "No Backup" command has been used, the "partial" segment is saved, since it has been written "over" the original.

4.18 Two Windows

Using AMIS it is possible to display two windows on the screen simultaneously. The screen is divided by a horizontal line in such cases. The area above the line is the first window, while the area below the line comprises the second. The two windows can display text from separate buffers independently, or they can display different text sections from the same buffer.

The following commands are used in this procedure:

Table 4-2. Commands for Window Handling

<**CTRL-X**> <**1**>	One window
<**CTRL-X**> <**2**>	Two windows, work point in second window
<**CTRL-X**> <**3**>	Two windows, work point in first window
<**CTRL-X**> <**O**>	Switches to the other window
<**CTRL-X**> <**^**>	Makes window in use bigger
<**CTRL-ESC-V**>	Leafs forward in the second window

<**CTRL-X**> <**1**>	presents the display in one window.
<**CTRL-X**> <**2**>	presents the display in two windows. The work point is placed in the second window. In order to display two different segments in the two windows, the command <**CTRL-X**> <**2**> is given, followed by <**CTRL-X**> <**CTRL-F**>. The latter command is used to switch to the buffer which contains the desired segment. If the user switches over to one window, the connection between the second window and its current buffer remains. If the command <**CTRL-X**> <**2**> is given again, this buffer will be presented immediately.
<**CTRL-X**> <**3**>	presents the display in two windows. The work point is placed in the first window.
<**CTRL-X**> <**O**> (Other Window)	is used to switch the work point back and forth from one window to the other. The erase buffer and Q-registers are common to all of the buffers. That is why it is easy to copy and move text sections between different buffers. The two windows display the handing activity in both buffers simultaneously. The procedure is as follows: Text can be removed from one buffer (to the erase buffer or a Q-register). The command <**CTRL-X**> <**O**> then moves the work point to the other buffer. The text can then be inserted from the erase buffer or the Q-register by using the command <**CTRL-Y**>.
<**CTRL-X**> <**^**>	This command increases the size of the window in which the work point is located.
<**CTRL-ESC-V**>	is used to leaf forward through the text in the window in which the work point is not located, at the rate of one page per time the command is given.

4.19 Command Macros

When the same sequence of commands is to be repeated many times, the commands can be written in as a command macro. A command macro starts with a single command. The commands in the command macro are executed in the order in which they were written in.The following commands are used to handle command macros:

<**CTRL-X**> <**(**>	Introduces a command macro
<**CTRL-X**> <**)**>	Concludes a command macro
<**CTRL-X**> <**E**>	Executes the command macro
<**CTRL-X**> <**Q**>	Sets conditions in a command macro

A command macro is defined in the following manner:

<**CTRL-X**> (<***command sequence***> <**CTRL-X**>)

<***command sequence***>	Consists of the desired commands.

The commands in the command sequence are executed at the same time as the command macro is defined. This provides a means of checking the sequence.

The command <**CTRL-X**> <**E**> starts the execution of the command macro. If a numeric argument is assigned to the command, the macro will be executed as many times as the numeric value indicates. An argument value of zero will cause the macro to be executed in a continuous loop, until some function, such as an unsuccessful search, cannot be performed.

The command <**CTRL-X**> <**Q**> (Query) makes it possible to set conditions in the macro pertaining to the macro’s continued execution. This command performs no function when the command macro is defined. During execution, this command will cause AMIS to ask whether or not to continue the execution. The following responses are used to control the continuation of the execution (several additional alternatives exist):

<**SPACE**>	Continues the execution
<**DEL**>	Terminates the present macro-loop
<**ESC**>	Terminates the macro
<**CTRL-R**>	Recursive editing. Return by using <**CTRL-Z**> <**Z**>

Both <**DEL**> and <**ESC**> terminate the macro, if no numeric argument has been assigned to the command <**CTRL-X**> <**E**>. If a numeric argument has been assigned, <**DEL**> terminates the execution of the present loop of the command sequence. The next loop in the macro is then started. <**ESC**> terminates the execution of the command macro, with or without a numeric argument.

4.20 Function Keys

F1	F2	F3	F4
SAVE Segment	Delete Character	Kill Line	Kill Region

Figure 4-1. Function keys used

* End of segment	- Forward Search	+ Unkill
7 Backward Word	8 Forward Word	9 Backward Search
4 Beginning of Line	5 End of Line	6 Beginning of Line
1 Previous Screen	2 Next Screen	3 Exch Point and Mark
0 Extended Command		. Set Mark

Figure 4-2. Keypad Layout

Those keys are only usable as described if NUMLOCK is active.

To use the cursor keys (those with the arrows), NUMLOCK must not be active, that is, the NUMLOCK indicator must be off, or, if NUMLOCK is on, use <SHIFT> and cursor keys.

Table 4-3. Function Keys

Function Key	**Equivalent Command Sequence**	**Explanation**
F1	<**CTRL-X**> <**CTRL-S**>	Save buffer on segment
F2	**<CTRL-D>**	Delete next character
F3	**<CTRL-K>**	Remove text up to end of line
F4	**<CTRL-W>**	Remove region
7	**<ESC> <B>**	Backward one word
8	**<ESC> <F>**	Forward one word
9	<**CTRL-R**>	Incremental backward search
-	<**CTRL-S**>	Incremental forward search
4	<**CTRL-A**>	Beginning of line
5	<**CTRL-E**>	End of line
6	<**ESC**> <**<**>	Beginning of segment
*	<**ESC**> <**>**>	End of segment
1	<**ESC**> <**V**>	One screen page backward
2	<**CTRL-V**>	One screen page forward
3	<**CTRL-X**> <**CTRL-X**>	Switch places with mark
0	<**ESC**> <**X**>	Named command
+	<**CTRL-Y**>	Recreate text last removed
←	<**CTRL-B**>	Backward one character
→	<**CTRL-F**>	Forward one character
↓	<**CTRL-N**>	Down one line
↑	<**CTRL-P**>	Up one line

4.21 List of Selected Commands

Table 4-4. List of Commands

Prefix characters	<CTRL-U> <CTRL-> <ESC> <CTRL-Z> <CTRL-X> <ESC> <X>	Numeric argument CTRL-prefix ESCAPE (META) CTRL-META-prefix C-X prefix Named commands
Multiple buffers	<CTRL-X> <CTRL-B> <CTRL-X> <B>	List available buffers Switches to another buffer
Multiple windows	<CTRL-X> <1> <CTRL-X> <2> <CTRL-X> <3> <CTRL-X> <0> <CTRL-X> <^> <CTRL-ESC-V>	One window Two windows, changes Two windows, remains Switch to other window Enlarges window in use Leaf through text in other window
Segments	<CTRL-X> <CTRL-S> <CTRL-X> <CTRL-V> <CTRL-X> <CTRL-F> <CTRL-X> <CTRL-W> <ESC> <X>	Save buffer in segment Read segment into buffer Find a segment in any buffer Display buffer for any segment Insert segment
Upper- and lower-case letters	<ESC> <U> <ESC> <L> <ESC> <C> <CTRL-X> <CTRL-U> <CTRL-X> <CTRL-L>	Change next word to upper-case Change next word to lower-case Change first letter of next word to upper-case Change all letters in region to upper-case Change all letters in region to lower-case
Comment handling	<ESC> <--->; <CTRL-ESC> <--->; <CTRL-X> ;	Insert comments after line Remove comments Set desired column for comments
Intended text and margin setting	<CTRL-I> <CTRL-J> <ESC> <^> <CTRL-X> <CTRL-I> <CTRL-ESC-\> <CTRL-X> <F> <CTRL-X> <.> <ESC> <Q> <ESC> <G>	Indent line “appropriately“ Same as RETURN followed by C-I Link line with preceding line Move text in region sideways Indent all lines in region Set right-hand margin Set margin string Applies margins to a paragraph Applies margins to a region

Table 4-4. List of Commands (Continued)

Mode	<ESC> <X> Text <ESC> <X> Fundamental <ESC> <X> Auto <ESC> <X> Overwrite <ESC> <X> Keypad	Text Mode Fundamental Mode Auto Fill Mode on/off Overwrite Mode on/off Keypad Mode
Moving the work point	<CTRL-F> <CTRL-B> <ESC> <F> <ESC> <B> <CTRL-P> <CTRL-N> <CTRL-A> <CTRL-E> <ESC> <A> <ESC> <E> <ESC> <[> <ESC> <]> <CTRL-V> <ESC> <V> <CTRL-X> <[> <CTRL-X>] <ESC> <<> <ESC> <>> <ESC> <ENTER>	Forward by character Backward by character Forward by word Backward by word Upward by line Downward by line Beginning of line End of line Backward one sentence Forward one sentence Beginning of paragraph End of paragraph Forward one screen page Backward one screen page Beginning of page End of page Beginning of entire text End of entire text Beginning of line
Search and replace	<CTRL-S> <CTRL-R> <ESC> <%>	Incremental forward search Incremental backward search Selective replacement
Escape and return	<CTRL-G> <CTRL-X> <CTRL-Z> <CTRL-]>	Terminate Escape AMIS Escape AMIS level
Help commands	<CTRL-_> <CTRL>->_<C> <CTRL-_> <D> <CTRL-_> <W> <CTRL-_> <A>	Help command Character Describe Where is Apropos

Table 4-4. List of Commands (Continued)

Erasing and moving text	<CTRL-D> <DEL> <ESC> <D> <ESC>-<DEL> <CTRL-K> <CTRL-W> <ESC> <K> <CTRL-X> <DEL> <CTRL-Y> <ESC> <Y> <ESC> <W> <ESC> <\> <CTRL-X> <CTRL-O> <CTRL-X> <X> q <CTRL-X> <G> q <CTRL-T> <ESC> <T> <CTRL-X> <CTRL-T> <CTRL-X> <T> <ESC> <^>	Erase next character Erase preceding character Remove next word Remove preceding word Remove text up to end of line Remove region Remove text to end of sentence Remove text from beginning of sentence Recreate text last removed Replace last removed text with older text Copy region to erase buffer Erase blank character Erase empty lines Copy region to Q-register q Insert text from Q-register q Two characters switch places Two words switch places Two lines switch places Regions switch places Link line to the preceding line
Command macros	<CTRL-X> <(> <CTRL-X> <)> <CTRL-X> <E> <CTRL-X> <Q>	Begin definition of command macro End definition of command macro Execute command macro Set conditions in command macro

Chapter 5 KERMIT

5.1 Introduction

KERMIT is a public protocol to transfer of files between various computers via ordinary terminal line. Programs implementing the KERMIT protocol have been developed by many people and organizations for a large number of different computers and operating systems.

The KERMIT protocol and some KERMIT implementations have been developed by Frank da Cruz et al. at the Columbia University Center for Computing Activities, New York 10027.

KERMIT version A.0/1 for Advant Station 100 Series Engineering Station (AS 100ES) is based on the KERMIT Protocol Manual, 6th edition, June 1986. For information about how to order KERMIT for various computer systems, write to:

KERMIT Distribution

Columbia University Center for Computing Activities

612 West 115th Street

New York, N.Y. 10025

A KERMIT distribution is also available from DECUS (Digital Equipment Corporation Users Society) for a small handling fee. Because KERMIT is free software, it may be freely copied and is available from a number of other sources for free or for a small handling fee.

This chapter primarily describes the Advant Station 100 Series Engineering Station KERMIT. The KERMIT running on MasterAid 720 is operated under “remote control” from the Advant Station 100 Series Engineering Station KERMIT, and thus very few commands need be given directly to the KERMIT on MasterAid 720.

For many types of computers and operating systems (e.g. VAX/VMS) more than one implementation of KERMIT exist. However, the KERMIT protocol supports a wide range of optional features including the ability to transfer files with contents other than text but only a few KERMIT programs can handle all the types of files used on ABB MasterAid systems.

Only the KERMIT program on Advant Station 100 Series Engineering Station can handle all the types of files which use the file system on the engineering station in the normal manner (it can not handle dump files which use entire disks without regard to the file and directory structure).

Only the special KERMIT program supplied with MasterAid 720 has all the functions needed for transfer of text files, binary files and load modules to and from a Advant Station 100 Series Engineering Station.

Text files, such as PC and data base source code may be freely transferred using any available KERMIT program.

5.2 Transferring Files

The following is a brief description of how to transfer files using KERMIT on the AS 100ES and the MasterAid 720. File transfer between AS 100ES and MasterAid 720 can be done in two ways:

- Start KERMIT on AS 100ES and then start KERMIT on VAX as server (described in this manual)
- Start KERMIT on VAX and then start KERMIT on AS 100ES as server (described in MasterAid 720 documentation).

5.2.1 Setting up the Connection

5.2.1.1 Preparing the Advant Station 100 Series Engineering Station

Connect the X4 connector on the rear of AS 100ES to an ordinary serial asynchronous terminal line (RS232C standard) connected to a VAX/VMS system with MasterAid 720.

Assuming the engineering station is already loaded with the AS 100ES Software, give the command KERMIT.

```
*KERMIT
```

The KERMIT program announces itself:

```
AS 100ES KERMIT version A.0/1
Device used for transfers is DB-25-Connector (X4).
If errors appear Extra 1 is equivalent to X4.
KERMIT-AS100>
```

Now you may enter KERMIT commands. To get some information about available commands, give the command HELP.

You may have to change the transmission parameters BAUD-RATE and/or PARITY before you can communicate with the MasterAid 720 system. Use the commands SET BAUD-RATE and SET PARITY to set these parameters so that they match the terminal connection on the MasterAid 720:

```
KERMIT-AS100>SET BAUD-RATE baudrate
```

The parameter baudrate is a number such as 4800 which matches the baud rate used on the MasterAid 720 system.

```
KERMIT-AS100>SET PARITY parity
```

The parameter parity is a keyword such as EVEN which matches the parity used on the MasterAid 720 system.

See the description of the command SET BAUD-RATE see Section 5.4.2.20 Set and the description of the command SET PARITY see Section 5.4.2.20 Set for a description of the available options.

5.2.1.2 Preparing the MasterAid 720

To transfer files, a KERMIT program must also be started on the MasterAid 720.

To establish a connection to the MasterAid 720, give the command CONNECT:

```
KERMIT-AS100> CONNECT
```

Now the AS 100ES will function as a (VT 100 compatible) terminal connected to the MasterAid 720 system. The MasterAid 720 will display its banner in the usual manner and prompt you to log in. Log in as usual.

When you have successfully logged in on the MasterAid 720, give the command KERMIT to start the KERMIT program on the MasterAid 720.

```
$ KERMIT
```

The KERMIT program on the MasterAid 720 announces itself and prompts for commands:

```
MasterAid 720 KERMIT version A.2/1
Default terminal for transfers is: _yyzz:
KERMIT-MA7>
```

If the prompt is not KERMIT-MA7>, but KERMIT-32> instead, the standard VAX/VMS KERMIT is installed. In this situation consult your system manager.

If non-text (binary) files are to be transferred, the command SET FILE TYPE BINARY must be given here. For text file, this is not necessary.

The KERMIT on MasterAid 720 must be prepared to accept commands from the KERMIT on your AS 100ES. To accomplish this, give the command SERVER:

```
KERMIT-MA7> SERVER
```

KERMIT on MasterAid 720 displays some text telling you to type an “escape sequence”. For KERMIT on AS 100ES, this means you should press the SHIFT and F4 keys simultaneously, release them and press the C key:

```
<SHIFT-F4> <C>
```

Now the KERMIT on the AS 100ES has left its terminal function, and prompts you for commands:

```
KERMIT-AS100>
```

At this point, you can start transferring files to and from the MasterAid 720.

5.2.2 Setting the File Type

Before you send or receive any files, you must determine the type of the file. AS 100ES KERMIT handles three types of files: ASCII (text) files, binary files and load modules. See Section 5.3 File Types for a detailed description of how to determine the type of a file.

If you intend to transfer only text files, you need not concern yourself with file types. AS 100ES KERMIT assumes all files are ASCII (text) files until you tell it otherwise.

If a non-text file is to be transferred, or if you have transferred non-text files previously and want to transfer text files without leaving KERMIT, the SET FILE TYPE command must be given to prepare KERMIT for the new file type.

Select the file type:

```
KERMIT-AS100>SET FILE TYPE file-type
```

The parameter file-type may be ASCII (for text files), BINARY or LOAD-MODULE (LOAD-MODULE is equivalent to BINARY when sending from a AS 100ES).

NOTE

The command SET FILE TYPE **must** be used each time a file of a different type is to be transferred. If several files are transferred using the same SEND or GET command with wildcard characters in the file name, the files must be of the same type.

This command should also be given to the KERMIT program in MasterAid 720. This may be done using the following sequence of commands:

```
KERMIT-AS100>FINISH
KERMIT-AS100>CONNECT

KERMIT-MA7> SET FILE TYPE file-type
KERMIT-MA7> SERVER
```

5.2.3 Transferring Files from AS 100ES to MasterAid 720

5.2.3.1 Transferring Single Files

To send a single file, use the command SEND:

```
KERMIT-AS100>SEND vol:segname.ex
```

The file on volume vol with name segname.ex will be sent to the current default directory on the MasterAid 720 (see command REMOTE CWD Section 5.4.2.17 Remote for a description of how to change the default directory).

Sometimes it may be useful to store the file on the MasterAid 720 under another name. This is accomplished by giving a second parameter to SEND:

```
KERMIT-AS100>SEND vol:segname.ex vaxfilename.vaxext
```

The file on volume vol with name segname.ex will be sent to the MasterAid 720 and stored as vaxfilename.vaxext. The second parameter may only contain a file name and a file type.

Some messages will be displayed on the screen while the file is being sent.

NOTE

The volume vol may be either USER or SRCE:

5.2.3.2 Transferring Multiple Files

Multiple files may be sent by including wildcard characters in the file name. See also the command TAKE Section 5.4.2.25 Take and Section 5.2.6.3 Example 3 for a description of how to use command files to transfer multiple files.

```
KERMIT-AS100>SEND vol:wildsegname.wildex wildvaxfilename
```

All files on the disk vol matching wildsegname.wildex will be sent and stored as wildvaxfilename. The second parameter is optional and may only contain a file name and a file type, at least one of which must be an asterisk "* ".

Wildcard characters allowed in parameters to the SEND command are "*" and "%". See the description of wildcard characters in Section 1.6.1.2 Wildcards

NOTE

All files sent with a single SEND command must be of the same type (ASCII, BINARY or LOAD-MODULE).

5.2.4 Transferring Files from MasterAid 720 to AS 100ES

5.2.4.1 Transferring Single Files

To receive a single file, use the command GET:

```
KERMIT-AS100>GET vaxfilespec vol:
```

The file specified by vaxfilespec will be transferred from the MasterAid 720 to the volume specified by vol: on the AS 100ES. The second parameter vol: may be left out if the file should be stored on the most recently used volume.

The parameter vaxfilespec may include a VAX/VMS device and/or directory name. The file will be stored on the Advant Station 100 Series Engineering Station with the same name and extension as on the MasterAid 720.

The file may be stored under a different name on the Advant Station 100 Series Engineering Station by specifying a full file name as the second parameter to GET:

```
KERMIT-AS100>GET vaxfilespec vol:segname.ex
```

The file specified by vaxfilespec will be stored on the volume vol as segname.ex on the Advant Station 100 Series Engineering Station.

5.2.4.2 Transferring Multiple Files

Multiple files may be received by including wildcard characters in the file name. See also the command TAKE Section 5.4.2.25 Take and Section 5.2.6.3 Example 3 for a description of how to use command files to transfer multiple files.

```
KERMIT-AS100>GET wildvaxfilespec vol:
```

All files matching wildvaxfilespec will be transferred from the MasterAid 720 to the volume vol on the Advant Station 100 Series Engineering Station.

Wildcard characters allowed in parameters to the GET command are all wildcard character allowed by VAX/VMS:

“*”	Matches any number of characters.
“%”	Matches a single character.
“...”	May be used in directory specifications to indicate searching downward through all directories below a specified directory level.

Wildcard characters in the GET command may be used according to the rules for specifying VAX/VMS file names.

It is possible to change a part of the name of the received files by specifying a segment name and a segment extension as the second parameter. However, at least one of the segment name and the segment extension must be replaced by an asterisk “*”.

```
KERMIT-AS100>GET wildvaxfilespec vol:*.newex
```

NOTE

All files sent with a single SEND command must be of the same type (ASCII, BINARY or LOAD-MODULE).

5.2.5 Closing the Connection

When you have finished transferring files, you should stop the KERMIT program on the MasterAid 720 and logout. This can easiest be done by giving the command BYE:

```
KERMIT-AS100>BYE
```

Now the KERMIT on the MasterAid 720 has logged itself out and you have at the same time left the KERMIT on the Advant Station 100 Series Engineering Station. You are back at the Advant Station 100 Series Engineering Station prompt:

```
*
```

5.2.5.1 Using the AS 100ES as a Terminal

If you wish to do some more work on the MasterAid 720 system using the Advant Station 100 Series Engineering Station as a terminal, you could have used the KERMIT command FINISH instead of the BYE command. This shuts down the KERMIT SERVER function on the MasterAid 720, and allow you to use the KERMIT CONNECT command to get back to the MasterAid 720:

```
KERMIT-AS100>FINISH
KERMIT-AS100>CONNECT
```

When you get the prompt from the KERMIT on the MasterAid 720, you may give the KERMIT command EXIT to get to VAX/VMS command level if you wish to give some ordinary VAX/VMS commands. Remember to restart KERMIT again on the MasterAid 720 if you wish to transfer more files.

```
KERMIT-MA7> EXIT          (to the MasterAid 720)
$
```

When you have finished working on the MasterAid 720, logout and escape back to the Advant Station 100 Series Engineering Station KERMIT using the <SHIFT-F4> <C> sequence. If you have not started the KERMIT SERVER function on the MasterAid 720, you may leave the Advant Station 100 Series Engineering Station KERMIT by just giving the command EXIT:

```
$ LOGOUT                  (to the MasterAid 720)

<SHIFT-F4> <C>            (to the AS 100ES)

KERMIT-AS100>EXIT

*
```

5.2.6 Examples

All text input by the user in the examples is shown in boldface characters.

5.2.6.1 Example 1

Example 1 shows how to transfer the load module AAAA00A0.LO from a MasterAid 720 system to a Advant Station 100 Series Engineering Station, storing the module as AAAA00A0.CP on the volume SRCE: in the AS 100ES.

```
*KERMIT
AS 100ES KERMIT version A.0/1
Device used for transfers is DB-25-Connector (X4).
If errors appear Extra 1 is equivalent to X4.
KERMIT-AS100>CONNECT
Connecting to remote host.
Press <SHIFT-F4> <C> to return to AS 100ES.
```

(The MasterAid 720 prompts for user name and password. Log in as usual.)

```
...
```

(The MasterAid 720 prompts for commands.)

```
$ KERMIT
MasterAid 720 KERMIT version A.2/1
Default terminal for transfers is: _TTA7:
KERMIT-MA7> SET FILE TYPE BINARY
KERMIT-MA7> SERVER
KERMIT server running on VAX/VMS host. Please type your escape
sequence to your local machine. Shut down the server by typing
the KERMIT BYE command on your local machine.

<SHIFT-F4> <C>
Connection closed, back at AS 100ES.
KERMIT-AS100>SET FILE TYPE LOAD-MODULE
KERMIT-AS100>GET <-.LOAD>AAAA00A0.LO SRCE:AAAA00A0.CP
```

(The file is received.)

```
...
KERMIT-AS100>BYE
*
```

(Back at Advant Station 100 Series Engineering Station command level.)

(The remote KERMIT on MasterAid 720 logs out at the BYE command.)

5.2.6.2 Example 2

Example 2 shows how several files of different types may be transferred between a MasterAid 720 and a Advant Station 100 Series Engineering Station.

```
*KERMIT
AS 100ES KERMIT version A.0/0
Default device for transfers is DB-25-Connector (X4)
KERMIT-AS100>CONNECT
Connecting to remote host. Press <SHIFT-F4> <C> to return to
AS 100ES.
...
```

(The MasterAid 720 prompts for user name and password. Log in as usual.)

```
...
```

(The MasterAid 720 prompts for commands.)

```
$ KERMIT
MasterAid 720 KERMIT version A.2/1
Default terminal for transfers is: _TTA7:
KERMIT-MA7> SET FILE TYPE BINARY
KERMIT-MA7> SERVER
KERMIT server running on VAX/VMS host. Please type your escape
sequence to your local machine. Shut down the server by typing
the KERMIT BYE command on your local machine.
```

(The user presses SHIFT and F4 simultaneously followed by a C.)

```
<SHIFT-F4> <C>
Connection closed, back at AS 100ES.
KERMIT-AS100>SET FILE TYPE LOAD-MODULE
KERMIT-AS100>GET *.LO SRCE:*.CP
```

(All load modules in def. directory.)

(The files are received.)

```
 ...
KERMIT-AS100>SEND SRCE:ABCD05B7.CP
```

(Send one command processor.)

(The file is sent.)

```
...
KERMIT-AS100>SET FILE TYPE BINARY
KERMIT-AS100>GET <-.PCDATA>FRED*.AD SRCE:
```

(The files are received.)

```
...
KERMIT-AS100>FINISH
KERMIT-AS100>CONNECT
KERMIT-MA7> SET FILE TYPE ASCII
KERMIT-MA7> SERVER
KERMIT server running on VAX/VMS host. Please type your escape
sequence to your local machine. Shut down the server by typing
the KERMIT BYE command on your local machine.
```

```
<SHIFT-F4> <C>
Connection closed, back at AS 100ES.
KERMIT-AS100>SET FILE TYPE ASCII
KERMIT-AS100>GET BLURBX01.AA
```

(Receive one text file.)

(The file is received.)

```
...
KERMIT-AS100>SEND GEORGE*.AA
```

(Send several text files.)

(The files are sent.)

```
...
KERMIT-AS100>BYE
*
```

5.2.6.3 Example 3

Example 3 shows how a command file is used with the command TAKE to transfer several files between a Advant Station 100 Series Engineering Station and a MasterAid 720.

Note how the command REMOTE CWD is used to select the directory on the MasterAid 720 from which the files are to be fetched, and how the command CWD is used to select the disk volume on the Advant Station 100 Series Engineering Station on which the files are to be stored. It is useful not to specify volume names in KERMIT command files, to make the command files more general.

Figure 5-1 shows the contents of the command file GETPC2.KF on the volume SRCE:

```
!This is a KERMIT command file to fetch all segments for PC program
!PC2 to the current volume.
!88-04-01 AUT/KP, Mary Smith

REMOTE CWD <.PCSOURCE>                          ! Change directory
SET FILE TYPE ASCII
GET IEXY*.AA                                    ! Get source code files
REMOTE CWD <.PCDUMP>                            ! Change directory
GET IEZQ*.ID                                    ! Get some text files
```

Figure 5-1. Command file GETPC2.KF

```
*KERMIT
AS 100ES KERMIT Version A.0/1
Device used for transfers is DB-25-Connector (X4).
If errors appear Extra 1 is equivalent to X4.
KERMIT-AS100>CONNECT
Connecting to remote host. Press <SHIFT-F4> <C> to return to
AS 100ES.
...
```

(The MasterAid 720 prompts for user name and password. Log in as usual.)

```
...
```

(The MasterAid 720 prompts for commands.)

```
$ KERMIT
MasterAid 720 KERMIT version A.2/1
Default terminal for transfers is: _TTA7:
KERMIT-MA7>SERVER
KERMIT server running on VAX/VMS host. Please type your escape
sequence to your local machine. Shut down the server by typing
the KERMIT BYE command on your local machine.

<SHIFT-F4> <C>
Connection closed, back at AS 100ES.
KERMIT-AS100>CWD USER
(the files are to be stored on USER:)
KERMIT-AS100>TAKE SRCE:GETPC2.KF
```

(Files are received.)

```
...
KERMIT-AS100>BYE
*
```

NOTE

For completely unattended operation the KERMIT BYE command may be included in the KERMIT command file as the very last line. Then you can give the TAKE command and go home. KERMIT will logout from the MasterAid 720 when the transfer is finished.

5.3 File Types

In Advant Station 100 Series Engineering Station and MasterAid 720 there are several types of files which must be handled in special ways when a file is to be transferred using KERMIT.

The KERMIT programs supplied with Advant Station 100 Series Engineering Station and MasterAid 720 can handle all file types used in the AS 100ES correctly. Other KERMIT implementations may not be able to handle certain MasterAid file types correctly without additional commands given to the KERMIT. Many KERMIT implementations cannot handle certain MasterAid file types at all.

The KERMIT command SET FILE TYPE is used to prepare KERMIT to handle files of a certain type. When KERMIT is started (using the KERMIT command), KERMIT assumes all files are of type ASCII (text). The KERMIT program supplied with MasterAid 720 makes an attempt to interpret the file structure when sending files from the MasterAid 720 to a Advant Station 100 Series Engineering Station and set the file type correctly.

NOTE

The following lists of file types are intended as examples only and are by no means exhaustive. Some dump commands produce several different types of files, where some may be text files and some may be binary files. The names of the dump files may be changed in future releases of Advant Station 100 Series Engineering Station and MasterAid 720 software.

5.3.1 ASCII (text) Files

Text files (also called ASCII files) may be transferred using KERMIT in either direction (to or from a Advant Station 100 Series Engineering Station) without any prior setup commands.

If files of other types have been transferred previously in the same session with KERMIT, the command SET FILE TYPE ASCII should be given before the SEND, GET or RECEIVE commands.

Text files do not require any specific version of KERMIT on a non-MasterAid system. You can transfer text files to and from almost any computer type using KERMIT without problems. This may be useful for backup purposes. However, be aware that Advant Station 100 Series Engineering Station and MasterAid 720 use the ASCII character set, and there may be problems with non-alphabetic characters and national characters like “÷”, “_” or “«” when transferring text files to machines which use other character sets.

Examples of text files are:

- PC and data base source code
- Command procedures and command procedure log files on AS 100ES.
- Any file created using the AMIS or EDIT commands on AS 100ES.
- “MESSAGE.ID” files produced by AS 100ES command DUTPT.

Certain text files on VAX/VMS systems may have an internal structure which is not directly visible when the file is TYPEd or PRINTed on the VAX but which is lost if the file is transferred using KERMIT. Examples of such files are .MEM files created by the VAX/VMS RUNOFF command and .PCL files created by the MasterAid 720 MAPC or MP100PC commands.

5.3.2 Binary Files

Binary files are all non-text files which should be transferred with the contents unchanged.

Binary files are sometimes referred to as “object code files” or “binary dump files”.

Examples of binary files are:

- Dump files produced by AS 100ES commands DUCON (file type “.CD”), DUAP (file type “.AD”), DUTDB (file type “.FD”), DURWM (file type “.DD”), DUTPT (file type “.AO”) and DUMM (file type “.PI”).
- Program image files produced by the MasterAid 720 command SGPOST.
- Data base dump files produced by the MasterAid 720 command MADB.

Examples of binary files which can not be transferred using KERMIT are:

- Files created using old versions of the MasterAid 215 DUMM command, except when the dump was made into a previously created segment of the correct size.
- Files created using new versions of the DUMM command and transferred using KERMIT, can be used by the LOMM command only if the file is recreated.

Advant Station 100 Series Engineering Station KERMIT always stores binary files as continuous segments on AS 100ES disks when the file type BINARY is selected. If file type SEQUENTIAL is selected, KERMIT creates sequential segments. Note that sequential segments may not be larger than 512 Kbyte and that sequential segments cannot be read by some Advant Station 100 Series Engineering Station load commands (such as LOMM).

The KERMIT supplied with MasterAid 720 stores binary files sent from Advant Station 100 Series Engineering Station KERMIT as sequential files with 256 byte fixed-length records. This file format is compatible with all MasterAid 720 software.

NOTE

Binary files transferred to a VAX/VMS system using any KERMIT which is neither the Advant Station 100 Series Engineering Station KERMIT nor the KERMIT supplied with MasterAid 720 will probably not be stored in the proper file format. Such files will not be usable as input to any MasterAid 720 software (with exception of the MP 100 Translator, in some cases).

5.3.3 Load Module Files

Load module files are Motorola load modules which should be expanded to a single continuous data area before they are stored in the Advant Station 100 Series Engineering Station. During the processing the header record of the file is removed.

An example of load module files is:

- AS 100ES command processors.

The load module file processing is only performed when a file is transferred to a Advant Station 100 Series Engineering Station. When load module files are transferred from a Advant Station 100 Series Engineering Station, they are treated as normal binary files.

NOTE

The LOAD-MODULE file type is specific for Advant Station 100 Series Engineering Station KERMIT. This file type is not defined in the standard KERMIT protocol. If load module files are transferred using a KERMIT other than the KERMIT supplied with MasterAid 720, the command SET FILE TYPE BINARY should be given to that KERMIT program.

5.3.4 Copying AS 100ES Volumes

When KERMIT is used to transfer files from one Advant Station 100 Series Engineering Station to another AS 100ES, file type BINARY may be used for all files (both text and binary). Thus, all contents of a Advant Station 100 Series Engineering Station volume may be transferred to another AS 100ES volume without regard to file type. In this case the file size does not have to be specified by the user.

5.3.5 Backing up AS 100ES Files to a Foreign System

When files are to be transferred from a Advant Station 100 Series Engineering Station to a foreign system such as VAX/VMS, file type BINARY may be used for all files if the files need not be used (typed, printed or edited) on the foreign system.

The files may be transferred back to the Advant Station 100 Series Engineering Station again using file type BINARY on the foreign KERMIT. On the Advant Station 100 Series Engineering Station KERMIT, file type SEQUENTIAL may be used for all files except those which are larger than 512 Kbyte or those which must be continuous segments.

5.4 Description of Commands

5.4.1 Command Summary

Advant Station 100 Series Engineering Station KERMIT is started using the command KERMIT. The program then prompts for additional commands.

The following is a summary of the commands available in Advant Station 100 Series Engineering Station KERMIT:

- Transferring files:
  - SEND, RECEIVE, GET
- Connecting to a remote host:
  - CONNECT, SET PARITY, SET SPEED
- Commands to the remote KERMIT:
  - BYE, FINISH, LOGOUT, REMOTE
- Command to get local information
  - LOCAL
- Setting transmission and file parameters:
  - SET FILE TYPE, SET FILE SIZE, SET PARITY, SET RETRY
- Setting default disk volume
  - CWD
- Interrupting CONNECT:
  - <SHIFT-F4> <C>
- Interrupting commands:
  - <CTRL-BREAK>
- Information:
  - HELP, STATUS, SHOW
- Executing command files:
  - TAKE
- Leaving the program
  - EXIT, QUIT, BYE

5.4.2 Detailed Command Descriptions

5.4.2.1 Command Syntax

KERMIT commands are entered after the prompt “KERMIT-AS100>” or are placed on one line in a KERMIT command file.

A KERMIT command is made up of a command verb, such as SEND, GET or EXIT, possibly followed by additional keywords or parameters.

Command verbs and keywords may be abbreviated to any length which makes them distinguishable from any other command verb or keyword valid for that position in the command line. For example, the command SET FILE TYPE LOAD-MODULE may be abbreviated to SET F T L.

Beware that later versions of KERMIT may include new commands and keywords, which may make old abbreviations ambiguous. For that reason, it is recommended that commands and keywords are not abbreviated when used in command files. All commands except CONNECT may be interrupted by pressing CTRL+BREAK.

5.4.2.2 Command Entry

KERMIT reads commands from the keyboard using a command line editor which permits the input command lines to be edited before the RETURN key is pressed. A KERMIT command line entered from the keyboard may be maximum 160 characters (2 lines on the screen) in length.

The following is a brief list of the keys available for editing a command line:

Cursor Movement

Key	**Action**
<←> or <CTRL-D>	Moves cursor to the left.
<→> or <CTRL-F>	Moves cursor to the right.
<CTRL-H>	Moves cursor to the start of the command line.
<CTRL-E>	Moves cursor to the end of the command line.
<DEL>	Deletes the character to the left of the cursor.
<F2>	Delete the character under the cursor.
<CTRL-J>	Deletes the word to the left of the cursor.
<CTRL-U>	Deletes all characters to the left of the cursor.
<F4>	Deletes all characters to the right of the cursor.

Recall previous Command:

Key	Action
<↑> or <CTRL-B> or <F3>	Recalls the previous command line. Maximum 10 lines are remembered.
<↓>	Recalls the next command line or an empty line.

Miscellaneous:

Key	Action
<CTRL-A>	Toggles between overstrike and insert mode.
<CTRL-R>	Rewrites the entire command line. Useful if a system message has overwritten the command input.
<F1>	Inserts a space at the cursor position.

5.4.2.3 Bye

Syntax:
`BYE`

Action:

Send a logout command to the remote KERMIT. When the remote KERMIT has acknowledged the command, the AS 100ES KERMIT exits to AS 100ES command level.

NOTE

Not all KERMIT implementations can handle the BYE command. You may have to use the FINISH command instead, and then use the CONNECT command to connect back to the remote computer and logout.

5.4.2.4 Connect

Syntax:
`CONNECT`

Action:

Establishes a connection to the remote computer. The engineering station will function as a VT 100-compatible terminal connected to the remote computer. The communications parameters parity and speed (baud rate) from the last SET PARITY and SET SPEED commands will be used.

To get back to KERMIT on Advant Station 100 Series Engineering Station, press the keys CTRL and F4 simultaneously, release them and press the C key.

5.4.2.5 CWD

Syntax:
`CWD volume`

Action:

The specified volume name on the Advant Station 100 Series Engineering Station will be used in subsequent KERMIT commands where the volume name is left out.

Example:

The following sequence of commands will receive the file GEORGE.AA and store it on the volume SRCE:

```
KERMIT-AS100>CWD SRCE
```

```
KERMIT-AS100>GET GEORGE.AA
```

5.4.2.6 Exit

Syntax:
`EXIT`

Action:

Terminates KERMIT on the Advant Station 100 Series Engineering Station and exits to AS 100ES command level.

EXIT is a synonym for the QUIT command.

NOTE

If a session on a remote computer has been started, you should use the BYE command or the FINISH command followed by CONNECT to logout the session on the remote computer before you give the EXIT command.

5.4.2.7 Finish

Syntax:
`FINISH`

Action:

Tells the remote KERMIT to exit from SERVER mode. After giving this command, you must connect back to the remote computer using the CONNECT command and either exit from the remote KERMIT and logout or start up the remote KERMIT server using the SERVER command.

5.4.2.8 Get

Syntax:
`GET` *`remote-filespec`*`[` *`local-filespec`*`]`

Action:

The file specified by **remote-filespec** will be transferred from the remote computer to the Advant Station 100 Series Engineering Station.

If **local-filespec** is specified, this will be the file specification on the Advant Station 100 Series Engineering Station.

The parameter local-filespec may contain a volume name, a segment name and a segment extension.

The segment name, segment extension or both may be specified as an asterisk “*”. The asterisk will be replaced by the corresponding part of the remote file specification.

The parameter remote-filespec is any file specification which is valid on the remote computer. It may contain device, directory and file name and type fields as they are used on the remote computer. This parameter is not checked by Advant Station 100 Series Engineering Station KERMIT in any way.

The parameter remote-filespec may contain wildcard characters (on VAX/VMS, these are “*”, “%” and “...”). If this is the case, the parameter local-filespec (if given) must have at least one of the segment name and the segment extension specified as an asterisk “*”.

When a file specification with wildcards matches more than one file on the remote computer, all matching files will be sent from the remote computer to the Advant Station 100 Series Engineering Station.

NOTE

> If wildcard are used to transfer more than one file, all transferred files **must** be of the same type.

Receiving binary files from non-MasterAid 720 KERMITs:

When GET receives a binary file from a KERMIT program which does not support sending a file length attribute before the file is sent, GET will query for the size of the file. Optionally, the SET FILE SIZE command may be given before the GET command to specify the size of the first file which GET receives. See the description of the command SET FILE SIZE Section 5.4.2.20 Set.

5.4.2.9 Local

Syntax:
`LOCAL`

Action:

The LOCAL command performs some utility functions on the local MasterAid. All LOCAL commands are also available as REMOTE commands when given to a Advant Station 100 Series Engineering Station KERMIT in SERVER mode.

5.4.2.10 Local Directory

Syntax:
`LOCAL DIRECTORY` *`volume:segment.extension`*

Action:

Displays a directory listing of the specified file(s). The segment name and extension may contain wildcard characters. The volume name may be replaced by a single asterisk “*”, in which case both on the Advant Station 100 Series Engineering Station will be searched for the specified files.

If no parameter is given, all files in the current default volume will be listed.

5.4.2.11 Local Space

Syntax:
`LOCAL SPACE` *`volume`*

Action:

Displays the free space on the specified volume. The volume name may be replaced by a single asterisk “*”, in which case the free space on both volumes of the Advant Station 100 Series Engineering Station will be displayed.

If no parameter is given, the free space on the current default volume will be displayed.

5.4.2.12 Local Status

Syntax:
`LOCAL STATUS`

Action:

Displays some information about the local KERMIT, e. g. version, currently used file type etc.

5.4.2.13 Local Type

Syntax:
`LOCAL TYPE` *`volume:segment.extension`*

Action:

Displays the contents of the specified file on the screen.

If the volume name is not specified, the current default volume is used. If the segment extension is not specified, the extension .KL is assumed (as in KERMIT log files).

5.4.2.14 Logout

Syntax:
`LOGOUT`

Action:

Tells the remote KERMIT to logout.

NOTE

> This will only work if you have given the remote KERMIT the SERVER command.

Some KERMIT program may not be able to understand a LOGOUT command. In this case you have to use the FINISH command to stop the KERMIT server and then CONNECT to the remote computer to logout the session.

5.4.2.15 Quit

Syntax:
`QUIT`

Action:

Terminates KERMIT on the Advant Station 100 Series Engineering Station and exits to AS 100ES command level. QUIT is a synonym for the EXIT command.

NOTE

> If a session on a remote computer has been started, you should use the BYE command or the FINISH command followed by CONNECT to logout the session on the remote computer before you give the QUIT command.

5.4.2.16 Receive

Syntax:
`RECEIVE[` *`local-filespec`***`]`**

Action:

The next file or group of files sent by the remote computer will be received. If **local-filespec** is specified, this will be the specification of the file on the Advant Station 100 Series Engineering Station.

The parameter **local-filespec** may contain a volume name, a segment name and a segment extension. The segment name, segment extension or both may be specified as an asterisk “*”. The asterisk will be replaced by the corresponding part of the remote file specification.

If wildcard characters have been specified in the SEND command on the remote computer, the parameter **local-filespec** (if specified) must have at least one of the segment name and the segment extension specified as an asterisk “*”.

NOTE

> This command is only needed when communicating with a remote KERMIT which lacks the SERVER function.

Receiving binary files from non-MasterAid 720 KERMITs:

When RECEIVE receives a file from a KERMIT program which does not support sending a file length attribute before the file is sent, RECEIVE will query for the size of the file. Optionally, the SET FILE SIZE command may be given before the RECEIVE command to specify the size of the first file which RECEIVE receives. See the description of the command SET FILE SIZE Section 5.4.2.20 Set.

5.4.2.17 Remote

The REMOTE command sends a command to the KERMIT on the remote computer. The KERMIT program on the remote computer must have been started, and the SERVER command must have been entered to the remote KERMIT.

5.4.2.17.1 Remote Generic-Command

Syntax:

```
REMOTE generic-command
```

Action:

Send a so-called generic command to the remote KERMIT server. The KERMIT protocol defines a number of generic commands which KERMIT server programs may or may not be able to handle. If the remote KERMIT does not recognize the command, you will see a message like “Unimplemented Server Command”.

The following commands are just a few of all possible generic KERMIT commands.

5.4.2.17.1.1 Remote CWD

Syntax:

```
REMOTE CWD remote-new-directory
```

Action:

Send a command to the remote KERMIT to change the current working directory to that specified by **remote-new-directory**. On VAX/VMS systems this is the equivalent to a SET DEFAULT command given at VAX/VMS command level.

5.4.2.17.1.2 Remote Director

Syntax:

```
REMOTE DIRECTORY[ remote-file-spec]
```

Action:

Send a command to the remote KERMIT to display a directory listing. If the optional parameter remote-file-spec is specified, only the files matching this file specification will be displayed.

5.4.2.17.1.3 Remote Type

Syntax:
```
REMOTE TYPE[ remote-file-spec]
```

Action:

Send a command to the remote KERMIT to display the contents of a text file.

5.4.2.17.1.4 Remote Who

Syntax:
```
REMOTE WHO
```

Action:

Send a command to the remote KERMIT to display a listing of users who are currently logged in.

5.4.2.17.2 Remote Host

Syntax:
```
REMOTE HOST host-command
```

Action:

Send the specified **host-command** to the remote computer for execution by the remote computers command processor.

Any output from the execution of the command will be sent back to the Advant Station 100 Series Engineering Station KERMIT and displayed on the AS 100ES screen.

If the remote computer is a VAX/VMS system, the **host-command** will be executed in a subprocess. Other computers may have other methods of executing the command.

NOTE

Some types of commands may not be possible to execute in this way. Commands which read more input from the terminal (on VAX/VMS for example editors, MAIL etc.) will not work.

5.4.2.17.3 Remote KERMIT

Syntax:
```
REMOTE KERMIT KERMIT-command
```

Action:

Send a KERMIT subcommand to the remote KERMIT server. This is typically a SET command to set file type. The command specified by **KERMIT-command** must be in the remote KERMITs syntax (as if it had been typed on the command line to the remote KERMIT program after the “KERMIT-xx>” prompt).

NOTE

The VAX/VMS KERMIT program supplied with MasterAid 720 can handle remote KERMIT commands, but other KERMIT implementations may not recognize such commands.

5.4.2.18 Send

Syntax:
`SEND local-filespec[ remote-filespec]`

Action:

The file specified by **local-filespec** will be sent to the remote computer. If **remote-filespec** is specified, this will be the specification of the file on the remote computer.

The parameter **remote-filespec** may only contain a file name and a file type. Either the file name or the file type or both may be specified as an asterisk “*”. The asterisk will be replaced by the corresponding part of the local file specification.

The **local-filespec** parameter may contain wildcard characters (“*” or “%”). When a file specification with wildcards matches more than one file on the Advant Station 100 Series Engineering Station, all matching files will be sent.

If local-filespec contains wildcard characters, the parameter remote-filespec (if given) must have at least one of file name and file type replaced by an asterisk “*”.

NOTE

If wildcard are used to transfer more than one file, all transferred files **must** be of the same type.

5.4.2.19 Server

Syntax:
`SERVER`

Action:

The Master Aid 200 KERMIT starts waiting for commands from a remote KERMIT program.

To stop the SERVER command, press CTRL-BREAK and then Q.

The MasterAid 200 KERMIT server can accept the following commands from a remote KERMIT:

GET	Receive file(s)
SEND	Send file(s)
REMOTE DIRECTORY	Display a directory listing
REMOTE KERMIT	Execute a KERMIT command
REMOTE SPACE	List free space on a disk
REMOTE STATUS	Display some information
REMOTE TYPE	List the contents of a file

The REMOTE KERMIT command accepts a few KERMIT commands such as SET FILE TYPE.

5.4.2.20 Set

5.4.2.20.1 Set Baud-Rate

Syntax:
```
SET BAUD-RATE speed
```

Allowed values for speed are one of: 300 1200 2400 4800 9600

Action:

Set the baud rate (transmission speed) of the Advant Station 100 Series Engineering Station communications device. The selected baud rate will be used in subsequent CONNECT commands and file transmissions.

This command is a synonym to the SET SPEED command.

NOTE

The selected baud rate must match that of the remote computer.

5.4.2.20.2 Set File Size

Syntax:
```
SET FILE SIZE number-of-blocks
```

Action:

Set the size in 1 Kbyte blocks of the next binary or load module file to be received using the GET or RECEIVE commands.

The specified file size is only valid for the next file received by a GET or RECEIVE command. Before each additional GET or RECEIVE command, you must give another SET FILE SIZE command to specify the size of the next file. If wildcard characters are used in file names to transfer more than one file, the file size specified by SET FILE SIZE is valid only for the first file received. For the following files, the GET or RECEIVE command will query for the file size.

If the SET FILE SIZE command has not been used to set the file size, the GET or RECEIVE commands will query for the file size.

This command is only needed when communicating with a remote KERMIT which does not support sending the file length attribute before the file itself. Currently the only known KERMIT which sends file length attributes is the KERMIT supplied with MasterAid 720.

To determine the size of a file stored on a VAX/VMS system, give the VAX/VMS command:

```
$ DIRECTORY/SIZE filespec
```

Divide the displayed size value by 2 (VAX/VMS specifies file sizes in 512-byte blocks) to get the number-of-blocks value to specify in the SET FILE SIZE command or as a response to a size query by the GET or RECEIVE commands.

5.4.2.20.3 Set File Type

Syntax:
`SET FILE TYPE` *`file-type`*

Allowed values for file-type are one of:

ASCII BINARY LOAD-MODULE SEQUENTIAL

Action:

Set the file type for subsequent SEND, GET or RECEIVE commands.

A file type of ASCII is the default and this is used for text files. A text file is any file you normally create using a text editor or a file containing PC or data base source code.

File type BINARY is used for all non-text files except load modules in Motorola format.

File type LOAD-MODULE is used for Motorola load modules which are to be expanded and have the header record removed when they are transferred to a Advant Station 100 Series Engineering Station using the GET or RECEIVE commands on the AS 100ES. When a file is transferred from a Advant Station 100 Series Engineering Station using the SEND command on the Advant Station 100 Series Engineering Station the file type LOAD-MODULE has the same effect as file type BINARY.

File type SEQUENTIAL may be used instead of BINARY when receiving binary files which are smaller than 512 Kbyte and which are to be used by applications which do not need continuous segments. File type SEQUENTIAL may use the space on a disk more efficiently (a sequential segment can use empty space between other segments on the disk even if these spaces are smaller than the sequential segment) and does not need to have the size specified in advance. However, some MasterAid 200 load commands may not be able to read sequential files. When a file is transferred from a MasterAid 200 using the SEND command on the MasterAid 200, the file type SEQUENTIAL has the same effect as file type BINARY.

NOTE

When transferring files from a Advant Station 100 Series Engineering Station to a remote computer where a KERMIT not supplied with MasterAid 720 is running, BINARY, LOAD-MODULE and SEQUENTIAL file types may not be handled correctly. When communicating with such a KERMIT program you may have to CONNECT back to the remote KERMIT and give a SET FILE TYPE command there in addition to the SET FILE TYPE command given to your Advant Station 100 Series Engineering Station KERMIT. Even if this is done, the file which is created on the remote computer may not be in the correct format. See the discussion of file types in Section 5.3 File Types.

5.4.2.20.4 Set Parity

Syntax:
`SET PARITY` *`parity-value`*

Allowed values for parity-value are one of:

EVEN ODD NONE

Action:

Set the parity of the Advant Station 100 Series Engineering Station communications device. The selected parity will be used in subsequent CONNECT commands and file transmissions.

Setting EVEN or ODD parity selects 7 bits data length and one parity bit.

Setting NONE selects 8 bits data length and no parity bit. Using SET PARITY NONE when possible is the most efficient way to transfer binary files.

NOTE
The selected parity must match that of the remote computer.

5.4.2.20.5 Set Retry

Syntax:
`SET RETRY` *`number`*

Action:

Specify the maximum number of times the Advant Station 100 Series Engineering Station KERMIT tries to send a packet before it gives up. If the SET RETRY command has not been given, AS 100ES KERMIT attempts to send a packet 5 times.

If the communications line you are using is very noisy, such as a long-distance telephone line, it may be useful to increase the number of retries allowed.

5.4.2.20.6 Set Speed

Syntax:
`SET SPEED` *`speed`*

Allowed values for speed are one of:

300 1200 2400 4800 9600.

Action:

Set the baud rate (transmission speed) of the Advant Station 100 Series Engineering Station communications device. The selected baud rate will be used in subsequent CONNECT commands and file transmissions.

This command is a synonym to the SET BAUD-RATE command.

NOTE
The selected baud rate must match that of the remote computer.

5.4.2.21 <CTRL-BREAK>

Syntax:
`<CTRL-BREAK>`

Press the keys CTRL and BREAK simultaneously.

Action:

Interrupts the execution of any command except CONNECT.

During communication with a remote KERMIT, an additional key has to be pressed to select the desired action. KERMIT will prompt for the key after CTRL-BREAK has been pressed.

CTRL-BREAK commands:

Key pressed	Action
Z	during file transfer, stops transmission of the current file and all the subsequent files.
X	During file transfer, stops transmission of the current file. Any subsequent files will be sent.
R	Retry. If possible, the interrupted command will be resumed.
Q	Quit. Stop immediately. During file transfer, the Z command should be used instead.

5.4.2.22 <CTRL-F4> <C>

Syntax:
`<CTRL-F4> <C>`

Press the keys CTRL and F4 simultaneously, then release them and press the C key.

Action:

Used during terminal emulation which has been started by the CONNECT command to end that mode of operation and to get back to the “KERMIT-AS100>” prompt.

After CTRL-F4 a few different functions are available:

CTRL-F4 commands:

Key pressed	Action
C	Close the connection and return to the “KERMIT-AS100>” prompt.
B	Send a long BREAK. A break signal which is 3.5 seconds long is sent. This may be useful for release of port selectors, hang-up of modems etc. for which the normal break (0.5 seconds) is too short.
E	Erase the screen. Also reset the character set, the scrolling region, tab stops etc.
H	Hardcopy. Sends a copy of the current screen contents to the printer.
0	Select ENGLISH character set.
1	Select SWEDISH character set.
?	Display a brief list of these command keys.

5.4.2.23 Show

Syntax:
`SHOW`

Action:

Displays current values of baud-rate, parity, file type and other parameters which may be set using the SET command.

5.4.2.24 Statistics

Syntax:
`STATISTICS`

Action:

Displays some statistics about the latest file transmitted. These statistics include the number of characters sent and received and some other data.

5.4.2.25 Take

Syntax:
`TAKE` *`command-segment-name`*

Action:

Starts taking commands from the KERMIT command file specified by command-segment-name. Commands are fetched from the command file until the end of the file is encountered.

If the segment extension is not specified, the extension “.KF” is assumed.

Commands in the command file are specified exactly as they are typed on KERMITs command line. Comments may be included in the command file by preceding them with an exclamation point “!”.

Command files used with the KERMIT TAKE command must only contain KERMIT commands.

NOTE

The KERMIT command CONNECT cannot be used in KERMIT command files.

5.5 Cables for KERMIT Connections

KERMIT communicates through the X4 connector of the Advant Station 100 Series Engineering Board. This 25-pole female D-type connector uses the V24 standard of electrical signal levels and pin assignments and may also be used with most RS232 equipment.

KERMIT is able to communicate with many different types of computers from many different manufacturers, which in turn may be connected through port selectors, modems etc.

ABB Automation and Drives supplies a cable for use with KERMIT (for ordering number, see the price list). The cable, when connected to Advant Station 100 Series Engineering Station, emulates a connection from a Digital Equipment Corporation VT 100 terminal to a modem, which will be suitable in most cases.

For those who wish to purchase the cable from a local supplier or manufacture it themselves, a description of the cable follows. Many cables which are used to connect an ordinary asynchronous terminal to a computer may also be used to connect a Advant Station 100 Series Engineering Station to a computer for use by KERMIT.

KERMIT does not use any of the handshaking signals in the connector. However, Advant Station 100 Series Engineering Station requires that certain signals be present in the connector which is inserted. This can be accomplished using a cable which is wired according to figure 5, 6 or 7. The cable shown in figure 6 is a simplified connection which makes do with a cheaper cable but it may not always work, depending on the type of modem used.

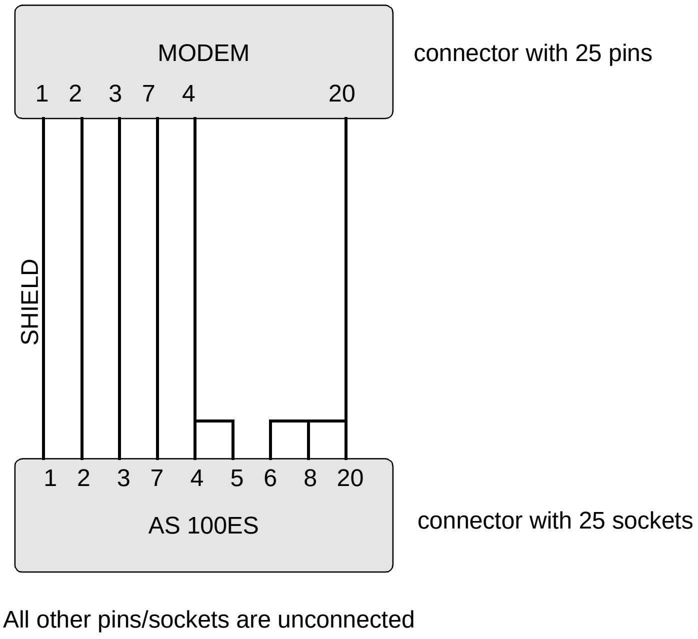
MODEM
connector with 25 pins
1
2
3
7
4
20
SHIELD
1
2
3
7
4
5
6
8
20
AS 100ES
connector with 25 sockets
All other pins/sockets are unconnected

Figure 5-2. Cable for Connecting AS 100ES to a Modem

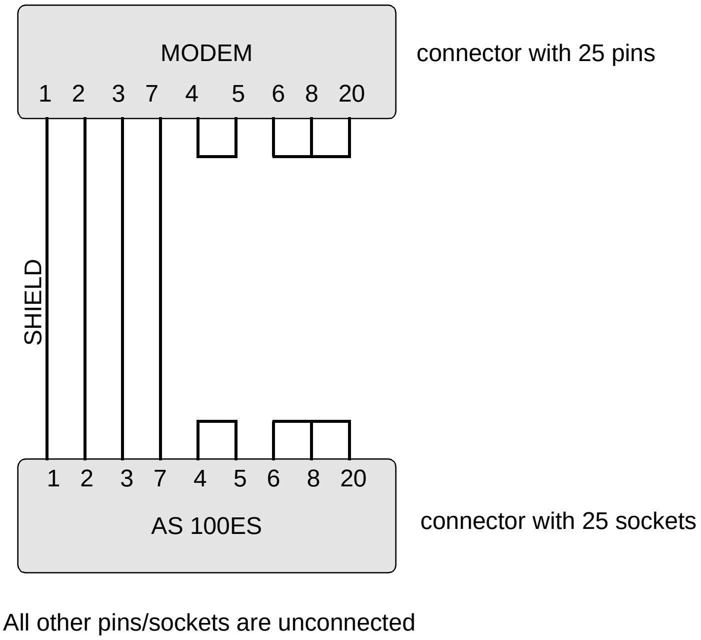
MODEM
connector with 25 pins
1
2
3
7
4
5
6
8
20
SHIELD
1
2
3
7
4
5
6
8
20
AS 100ES
connector with 25 sockets
All other pins/sockets are unconnected

Figure 5-3. Cable for Connecting AS 100ES to a Modem (simplified)

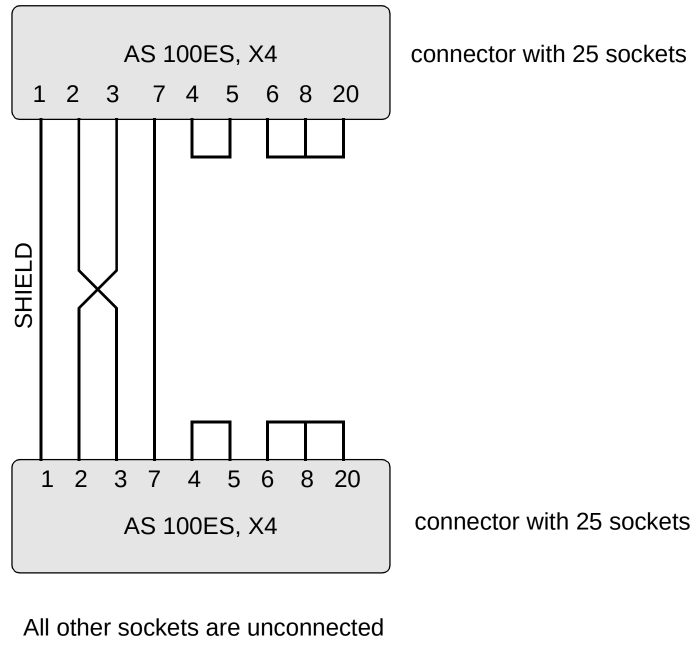
AS 100ES, X4
connector with 25 sockets
1 2 3 7 4 5 6 8 20
SHIELD
1 2 3 7 4 5 6 8 20
AS 100ES, X4
connector with 25 sockets
All other sockets are unconnected

Figure 5-4. Cable for Connecting AS 100ES to AS 100ES or Another Computer

If an error message from KERMIT is obtained which says something like “Connector not inserted?” the wiring inside the connector which is plugged in should be checked. Some of the connections between the pins 4-5 or 6-8-20 may be missing.

INDEX

A
Alternative choices 1-1
Arguments 1-1

B
Bold strings 1-1
Bold strings in screen dialogs 1-2
Brackets 1-1

C
Comma 1-1
Command File GETPC2 5-10
Command syntax 1-1

E
Enter key 1-1

I
Italic strings 1-1

K
Keyboard Interaction 1-1

N
Normal strings in screen dialogs 1-2

O
Options. See Arguments 1-1

P
Parameters. See Arguments 1-1

R
Right angle bracket 1-1

S
Screen dialogs 1-2
Semicolon 1-1
Slash 1-1
SRCE 1-2
Substitutions 1-1
Syntax 1-1
Switches. See Arguments 1-1

U
USER 1-2

V
Vertical bar 1-1

W
Wild cards
  %, * 1-3

3BSE 002 423R0001

August 1994